Bluetooth Cochlear

Jiawen Gu, 5130309763

2016. 06. 25

Overview

This report is mainly about the project we do this semester. That's the development of bluetooth cochlear. We cooperate with Nurotron (诺尔康公司) to associate bluetooth with traditional cochlear to enhance to provide better performance and user experience of artificial cochlear. Currently, we have finished the first step of our cooperation, the job we have done will be introduced in this report and the future work, as well as the next step of cooperation will be mentioned at the end.

Content

1. Background

1.1. Artificial Cochlear

Fig1. Extracorporeal Device

Fig2. Cochlear Implant

A traditional artificial cochlear is made up of two parts — a extracorporeal device and a cochlear implant. A extracorporeal device collects and processes environmental sound and transfer the encoded sound signal to the cochlear implant. The receiving coil of the implant receives the signal and control the electrodes according to preset program. Finally, the electrodes stimulate acoustic nerve ending and the user can hear something like the original sound.

What we do is actually like a controller of the artificial cochlear.

1.2. Bluetooth Chip - CSR8670

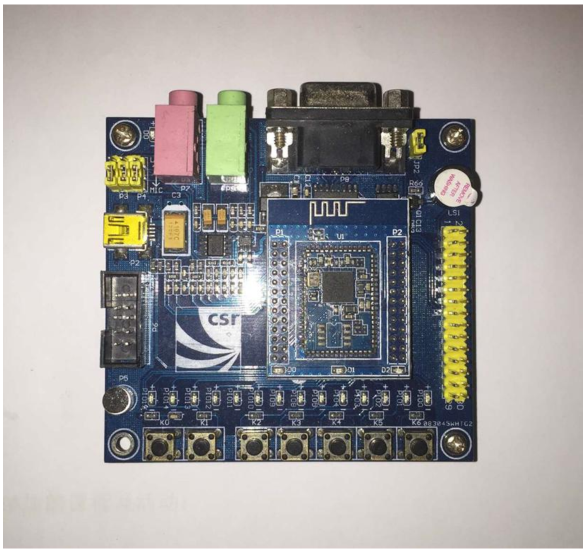


Fig3. CSR8670

The bluetooth chip we use is the CSR8670. It's widely used while developing bluetooth audio products. It supports Bluetooth basic rate / EDR and low energy connections. Also a lot of physical interface such as UART and USE, really suitable for development.

1.3. Control Protocol - GAIA

The control protocol supported by CSR8670 is GAIA—Generic Application Interface Architecture. It's established on a SPP connection and it implements an end-to-end,host-agnostic ecosystem supporting host application access to device functionality.

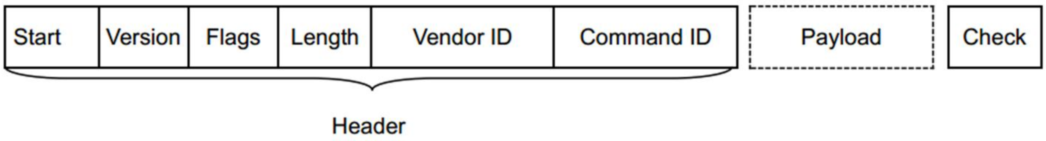
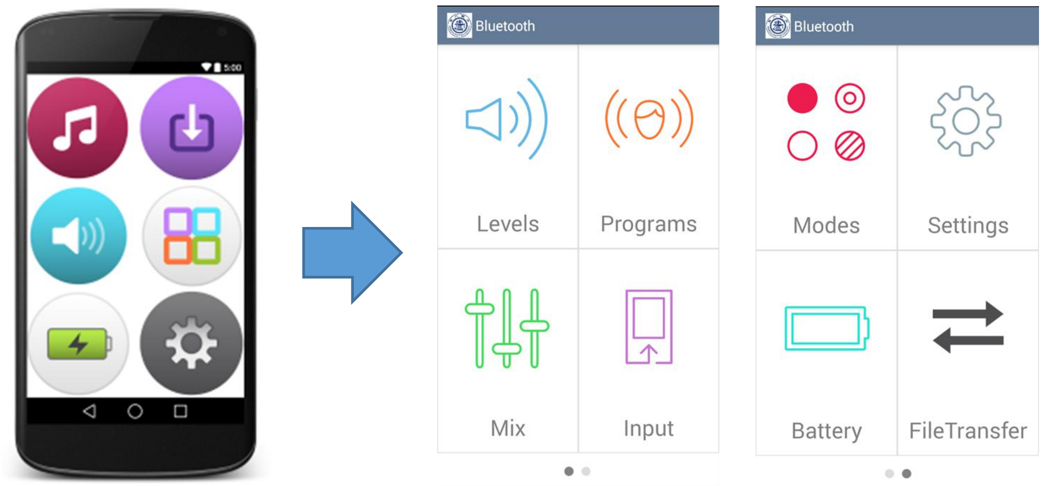


Fig4. GAIA Command Format

The format of a GAIA command is as shown above. The header is all defined by the protocol itself. What we need to provide is the payload varying from one command to another.

2. Project Progress

2.1. UI Optimization



The first thing we do is optimizing the UI of the Android APP. The left one is the protocol, and the right one is the new UI. Obviously, the new one looks much better.

2.2. Basic Functions

The Android app we build works as a controller of the artificial cochlear. We implement some basic functions as below:

- Paring
  We can pair with a bluetooth device inside our app. This is support by the Android API — *BluetoothAdapter* and *BluetoothDevice.* After pairing, we establish a SPP connection between android device and the bluetooth chip so that we can use GAIA command to control the bluetooth chip.

- Device status inquiry
  Currently, we can periodically query the battery and RSSI of the bluetooth chip. This is supported by the GAIA protocol.

- Remote volume control
  We can control the output volume of the bluetooth chip by sending a GAIA command from android device to the chip where the volume range from 0x00 to 0x0f.

Fig5. Pairing and Device status

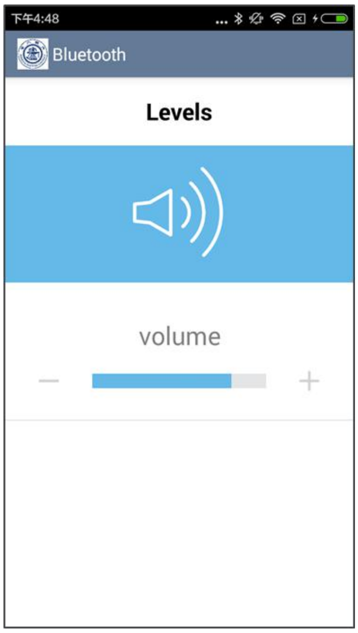


Fig6. Remote volume control

2.3. UART

We realize bidirectional data transfer through UART. For example, a PC can send a command to the bluetooth chip to control it. Meanwhile, the chip can dump data like script and files to PC.

In our project, we connect to the RS232 of the PC and realize the control of

LED on the development board via command sending through UART.

2.4. File Transfer Rate Test

When conducting DFU(Device Firmware Upgrade), we have to transfer the upgrade package to the bluetooth chip. So we are asked to test the file transfer rate between an android device and the bluetooth chip.

We send packets via GAIA commands with each packet carrying at most 250 bytes limited by the max payload of GAIA protocol. We implement both stop-and-wait and pipeline methods to test the transfer rate and get the maximal rate to be 2.5KB/s.

Undoubtedly, it’s very slow. So we check the source code on the bluetooth chip and read some documents about CSR8670. Finally, we conclude that the GAIA protocol is not the only protocol supported by the chip. So the bandwidth and computation resource allocated to it is limited. As a result, it can handle only one GAIA command at a time. Any packet received while processing will be received and demodulated but not handled. It will be dropped before it's delivered to application layer.

However, in case that the DFU packet is generally at size of several KB, this transfer rate can already meet the demand.

2.5. Device Loss Alarm

The extracorporeal device is very expensive and is put right behind the user’s ear. So there’s chance that the device fall down form ear and get lost. However, the cochlear implant can not work alone. It can be a great loss and very uncomfortable to the user.

Therefore, we come up with the idea of monitoring the RSSI of the bluetooth chip and set a threshold to trigger an alarm.

$$d = 10^{\frac{|RSSI| - A}{10n}}$$

The formula shows the theoretical relationship between RSSI and distance where A is the actual |RSSI| when d = 1m and n is the attenuation factor. It seems that we can estimate the distance according to this formula.

However, the RSSI actually fluctuates a lot. The following graph shows the distribution of |RSSI| when the bluetooth chip is 2 meters away from an android device.

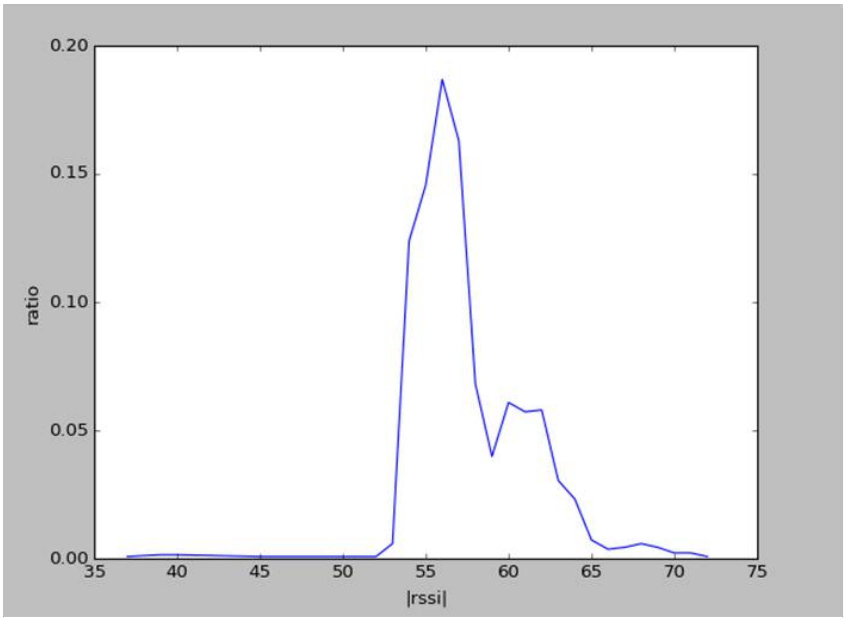


Fig7. |RSSI| distribution

We can see the value ranges from -72db to -37db and the average value is -57db. It's very unstable. So we cannot simply judge the distance according to one single RSSI. In the above graph, we can see most of RSSI concentrate on the range -65db to -53db. And as the distance increases, it becomes less and less possible to be greater than -53db. So we decide to use three continuous RSSI smaller than threshold to trigger an alarm. And to make it more sensitive, we don't set the threshold to be the average value, but the greatest RSSI with a relatively high possibility of appearance. For example, -54db in the above graph.

This is the interface when an alarm is activated. At that time, the cell phone will vibrate to alert the user. And there is an estimated distance according to the average of three latest RSSI to help the user to find the device.

3. Future Work

3.1. Scene Recognition

Generally, a artificial cochlear has different sound processing program in different scenes. For example, when the user is in a quiet condition, the corresponding program will lower the output volume of the implant. What we are going to do is implementing a learning based audio scene recognition and then the cochlear can automatically apply different sound processing programs to different scenes. It will make the artificial cochlear smarter and more convenient to use.

3.2. Noise Reduction

The interference of noise has always been a obstacle to artificial cochlear. If we can filter the noise of the environment sound, it will greatly improve the performance of artificial cochlea in noisy environment. However, this can be challenging.