Using Bluetooth Heart Rate monitor from Java ME

This article describes how to use a SportsTracker HRM2 Bluetooth (ST HRM2) belt from a Nokia Asha (or any other JSR-82 enabled) device.

24 Nov 2013

Note: This is an entry in the Nokia Asha Wiki Competition 2013H2.

Introduction

Heart Rate monitors are popular tools for sport enthusiasts and the Asha/S40 devices make excellent companion devices. They are relatively cheap and the battery lasts almost forever.

An exercise tracking application needs data and one such is the heart rate. There are various types of heart rate monitoring (HRM) belts available. This article describes how to use one such device, a SportsTracker HRM2 Bluetooth (ST HRM2) belt from a Nokia Asha (or any other JSR-82 enabled) device.

Note that using the belt involves two operations, connecting to the belt, and then reading data.

Connecting to the ST HRM2

The belt shows up as any other Bluetooth device, providing a SPP service. First you need to pair the belt with your device, just follow the instruction provided by the belt:

1. Make sure the Bluetooth dongle is charged
2. Put the belt on yourself
3. Moisturise the pads, water works
4. Enable Bluetooth on your Asha and go to the paring menu
5. Clip on the ST HRM2 dongle, it will power-on and be discoverable for about a minute or so
6. Scan for new devices on the phone, and when you see the ST HRM2, pair with it

Now you are ready to connect your device with the belt, (using of course the Bluetooth address of your own device) example code below:

```
 try {
    c = (StreamConnection) Connector.open("btspp://FF0BAE1CA3E0:1",
Connector.READ_WRITE, true);
    data = new DataInputStream(c.openInputStream());
    connected = true;
    listener.heartRateConnected();
    return true;
 } catch {
```

The next step is to read data from the belt and make sense of it.

Reading the data packets

ST HRM2 Protocol

The belt uses the standard Bluetooth Serial Port Profile (SPP), so it is easy to connect to and to read data from. Unfortunately, the format of the data-stream is not publicly documented and had to be reverse engineered.

When a connection has been established, the belt will immediately start to send out a data packets about once every second. By monitoring this data with a hex-dumper it was possible to figure out where the HR value is and how to parse it out from the byte-stream the belt sends.

The data packet contains:

1. 0xFA Start of data packet
2. Sequence number going up from 0x44 to 0x47

(C) Copyright Nokia 2014. All rights reserved.

3. Sequence number going down from 0xBB to 0xB8. The sum of fields 2 and 3 seem to always be 0xFF
4. 0x81 Seen to be 0x01 on first connect, but after that stays at 0x81
5. 0x?? Varies, most likely battery charge information, as it is large after charge and then goes down. Also seen as 0x53 on first connect, if field 4 is 0x01. 0x3C when fully charged.
6. 0x?? Heart rate measurement, the value we are looking for
7. 0x00 7-10 sometimes have values, sometime they 0x00
8. Padding of zeroes, amount varies
9. Packet ends with two bytes, most likely a checksum

More details about the reverse engineering of the protocol and the various data dumps used are available in a github repository.

Code to read the data

When reading the data packet we start by reading bytes until we find the start of packet marker (0xFA). Then we read the next two bytes and check that they add up to 0xFF. At this point we can be sure that we have found a valid data packet.

Next we read the next two bytes in the packet. The second byte is the battery level of the belt (the meaning of the first byte is unknown).

Then we read in one more byte, which is the heart rate. The data packet contains other bytes after the heart reate, but as we don't know what they mean we just start searching for the start of packet marker again.

The following method, does exactly this, one byte at a time:

```
private boolean readHRMData() {
        int i=-1;

        try {
            i=data.read();
        } catch (IOException e) {
            Log.loge("IOe", e);
            return false;
        }

        if (i==-1)
            return false;
        if (i==0)
            return true;

        switch (state) {
            // Looking for start message byte 0xFA
            case STATE_INITIAL:
                if (i==0xFA) {
                    // We got start marker, go to next state
                    state=STATE_MARK1;
                } else {
                    return true;
                }
                break;
            case STATE_MARK1:
                mark1=i;
                state=STATE_MARK2;
                break;
            case STATE_MARK2:
                mark2=i;
                if (mark1+mark2!=0xFF) {
                    state=STATE_INITIAL;
                    return true;
                }
```


(C) Copyright Nokia 2014. All rights reserved.

```
                state=STATE_UNKNOWN1;
                break;
            case STATE_UNKNOWN1:
                // 0x81 ?
                u1=i;
                state=STATE_BATTERY;
                break;
            case STATE_BATTERY:
                // Battery level ?
                if (u1==0x81 && i!=battery) {
                    battery=i;
                    listener.heartRateBattery(i);
                }
                state=STATE_HRM;
                break;
            case STATE_HRM:
                hr=i;
                listener.heartRate(hr);
                // Ok, we got what we need, wait for initial marker. We ignore anything
else for now
                state=STATE_INITIAL;
                break;
        }
        return true;
 }
```

A complete runnable example program is available on github. Note however, the example lacks a device chooser, the HRM belt Bluetooth address is hard-coded, you need to change it for your own device!

Other HRM belts

There are many other HRM belts available and most of the new ones are using Bluetooth 4.0 and are using the BT LE protocol. These belts are not compatible with any current Asha/S40 devices as none support Bluetooth 4.0 Low Energy devices.

All older belts, that use the SPP protocol, can most likely be used just fine. Some (Zephyr HxM) even have public documentation available. Unfortunately, the article author don't own any of these so support for those will be left as an exercise for the reader.

More information

- Protocol reverse engineering information
- Example application github repository
- Blog post 1 & Blog post 2

Summary

With this information it should be possible to create an application for monitoring the heart rate of the user, with a ST HRM2 belt and Asha device.

Version Hint

Windows Phone: [[Category:Windows Phone]]
[[Category:Windows Phone 7.5]]
[[Category:Windows Phone 8]]

Nokia Asha: [[Category:Nokia Asha]]
[[Category:Nokia Asha Platform 1.0]]

Series 40: [[Category:Series 40]]

(C) Copyright Nokia 2014. All rights reserved.

[[Category:Series 40 1st Edition]] [[Category:Series 40 2nd Edition]]
[[Category:Series 40 3rd Edition (initial release)]] [[Category:Series 40 3rd Edition FP1]] [[Category:Series 40 3rd Edition FP2]]
[[Category:Series 40 5th Edition (initial release)]] [[Category:Series 40 5th Edition FP1]]
[[Category:Series 40 6th Edition (initial release)]] [[Category:Series 40 6th Edition FP1]] [[Category:Series 40 Developer Platform 1.0]] [[Category:Series 40 Developer Platform 1.1]] [[Category:Series 40 Developer Platform 2.0]]

Symbian: [[Category:Symbian]]
[[Category:S60 1st Edition]] [[Category:S60 2nd Edition (initial release)]] [[Category:S60 2nd Edition FP1]] [[Category:S60 2nd Edition FP2]] [[Category:S60 2nd Edition FP3]]
[[Category:S60 3rd Edition (initial release)]] [[Category:S60 3rd Edition FP1]] [[Category:S60 3rd Edition FP2]]
[[Category:S60 5th Edition]]
[[Category:Symbian^3]] [[Category:Symbian Anna]] [[Category:Nokia Belle]]

(C) Copyright Nokia 2014. All rights reserved.