

HJ-BT-16G 蓝牙振动传感器说明书

产品简介

HJ-BT-16G 振动传感器是新一代基于蓝牙 5.2 的振动传感器，是理想的设备安全监测，预测性维护的得力助手。通过安装在设备上的 **HJ-BT-16G** 无线振动传感器，获得实时的加速度信号,并通过加速度计算得到速度（振动烈度）和位移等标准振动量，同时采集温度，进行无线通道传输。该传感器具有反应振动的多组特征值输出：三轴加速度，三轴速度，三轴位移，温度，倾斜角度（可反映传感器是否掉落等情况），设备转速（最强频谱频率以及幅值），能够充分反映振动状态。

产品特点：

无需开关机，通过蓝牙随时随地可进行无线唤醒链接数据传输

支持传统特征值以及波形数据传输

使用场合：

设备点检配套

设备振动监测，振动分析

性能指标

振动信号频响：

加速度 0Hz~2KHz；

速度 8Hz~1KHz；

位移 8Hz~1KHz；

采样速率 200Hz~8kHz 可设定；

测量通道： 3 轴；

温度测量-40℃ ~ 85℃，测量精度±0.5℃；

通讯方式 SRRC 认证 BLE5.2

传输有效距离 200m；

电池容量：电池单节 1 号锂亚电池（电量 19000mA/H）。

也可以选择 12V~24V 供电版本。

防护等级：IP67，表面耐腐蚀；

防爆等级：EXia II BT4；

安装方式：磁吸座/M8 螺纹/胶粘安装可选；

电池更换方式：上盖拧掉即可轻松更换

传感器输出：

三轴加速度值（±16G）；

三轴速度值(0-500mm/s)；

三轴位移值(0-10mm)；

三轴频谱最高频率以及幅度值

电池电压；

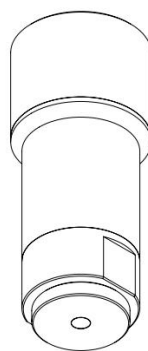
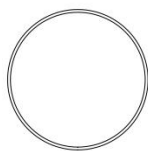
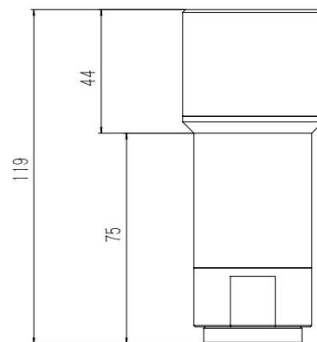
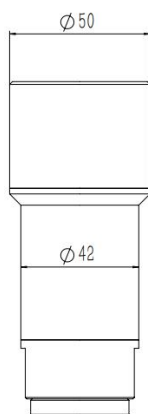
三轴重力矢量或倾斜度（可判断传感器位置是否有移动）；

加速度时间波形

结构尺寸

(单位 mm)

全密闭版 (高可靠性, 推荐选用) (单位 mm)



● 指示灯说明:

传感器上电时候灯会闪烁，之后灯灭进入低功耗状态随时等待主机连接。
在建立连接之后传感器灯会慢闪，就可以通过指令进行设对或者数据采集。

● 上电说明

1. 拿到传感器后，拧开上盖，箭头所指接口处（**!!! 注意：底部底座不可拧开，拧开传感器即损坏**）



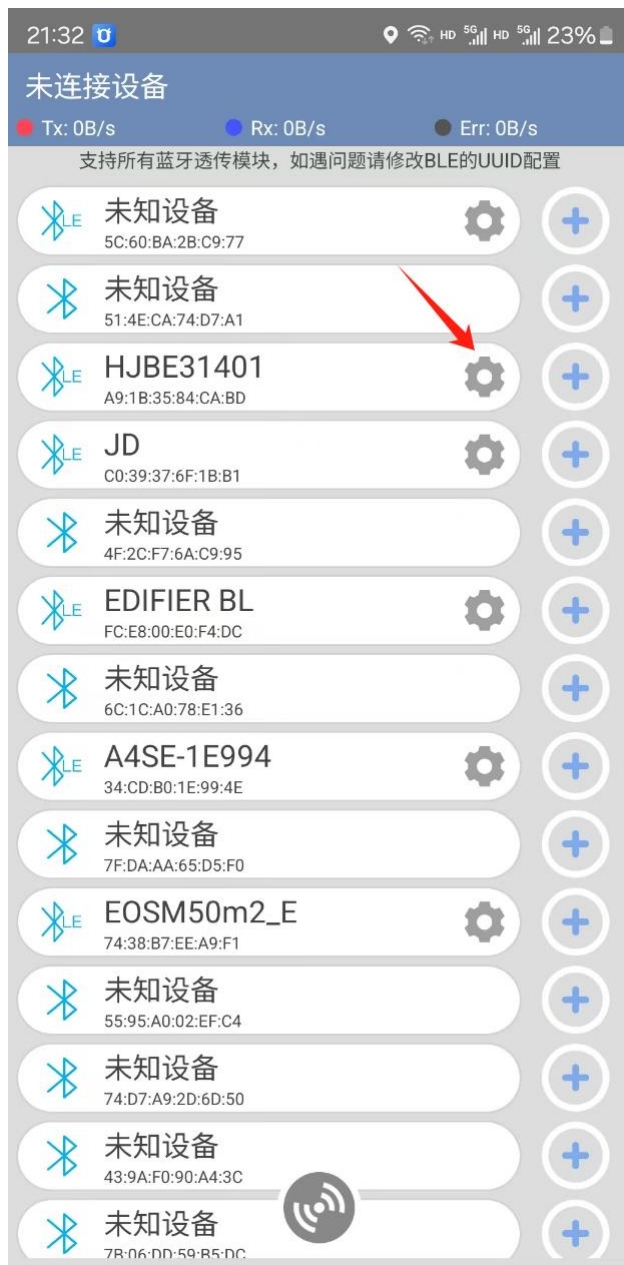
2. 将里面电池对插接头对插接上，**并将线整理到壳体四周，如下图所示。**
电池接上的时候可以看到传感器指示灯会点亮并闪烁，之后重新关闭。传感器进入待机状态，即可使用手机软件进行连接，交互。



3. 拧上盖子即可

使用蓝牙调试器测试过程说明

- 首先搜索蓝牙，找到 HJBE 开头的设备并点击设置符



- 严格按照图示进行配置



- 点击+, 看到变成 x 之后表示成功连接传感器

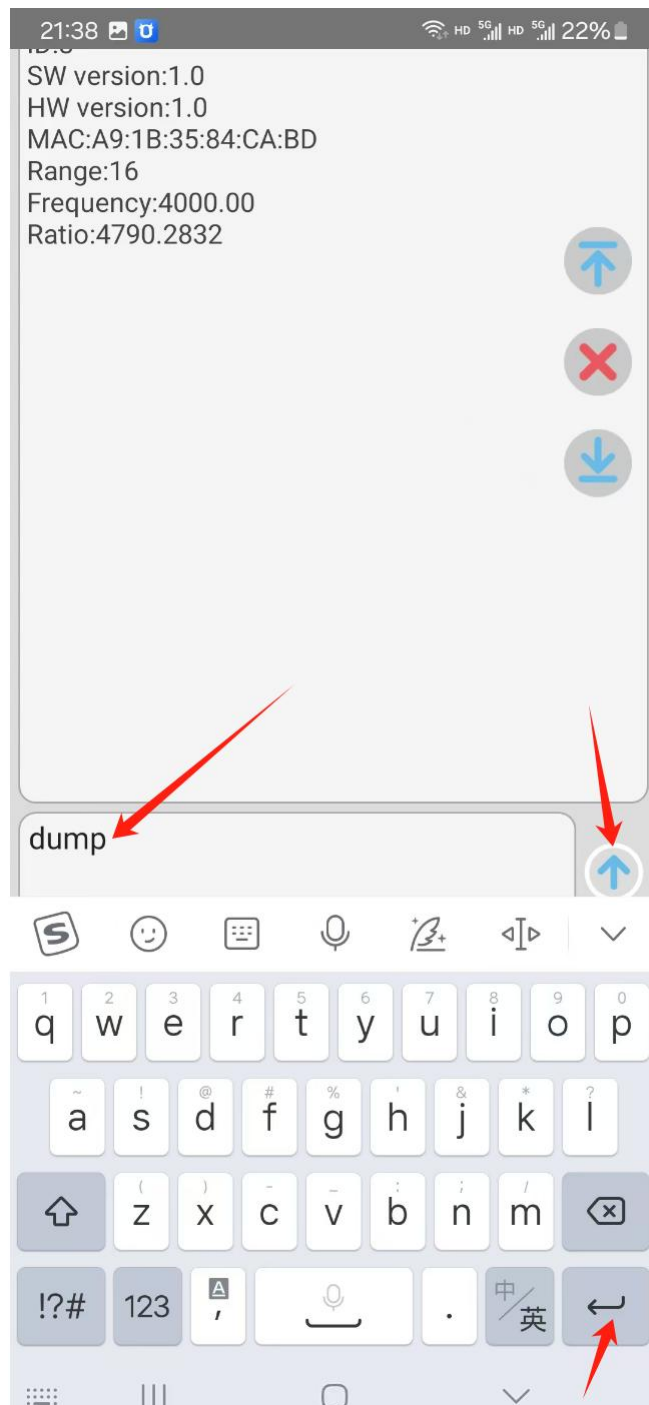


- 选择对话模式



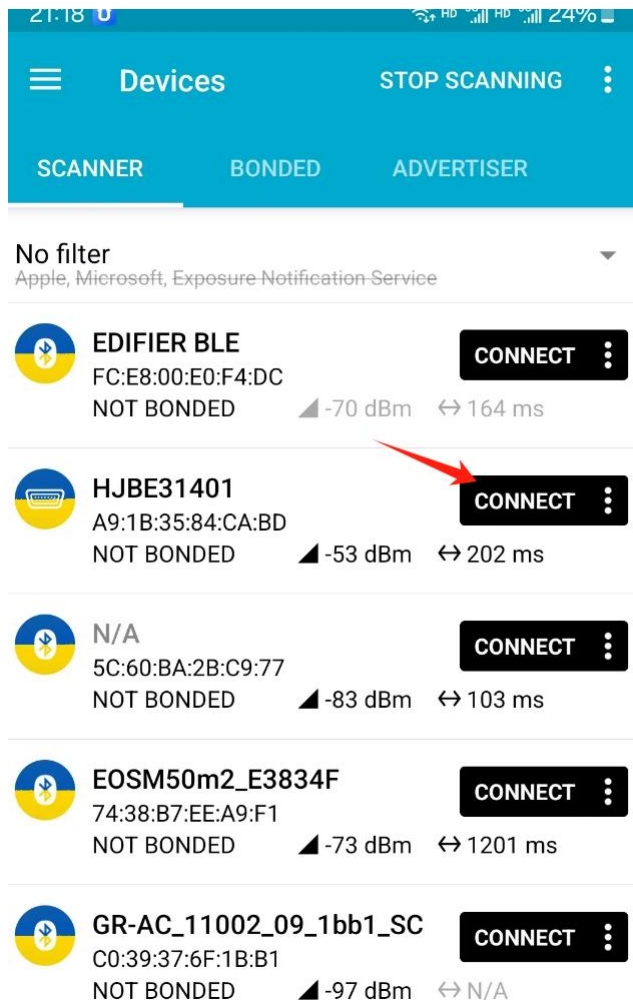
- 输入 dump 并回车键, 然后按发送按钮

看到传感器信息正常输出说明通讯正常



使用 nRF Connect 测试说明

- 选择 HJBE 开头的蓝牙设备，点击 CONNECT
若成连接传感器蓝牙，可看到传感器指示灯开始闪烁。



- 点击 UART SERVICE

☰

Devices

DISCONNECT

DFU

⋮

BONDED

ADVERTISER

HJBE31401
A9:1B:35:84:CA:BD

×

CONNECTED

NOT BONDED

CLIENT

SERVER

⋮

Generic Access

UUID: 0x1800

PRIMARY SERVICE

Device Information

UUID: 0x180A

PRIMARY SERVICE

Nordic UART Service

UUID: 6e400001-b5a3-f393-e0a9-e50e24dcca9e

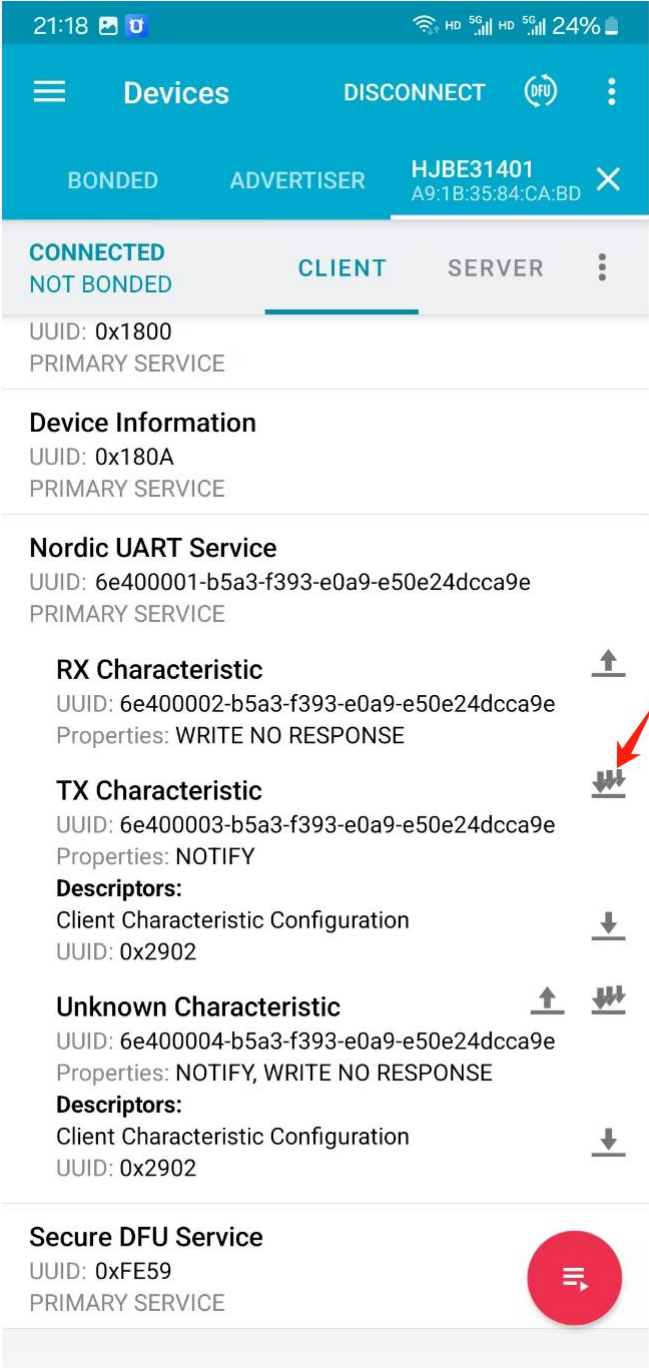
PRIMARY SERVICE

Secure DFU Service

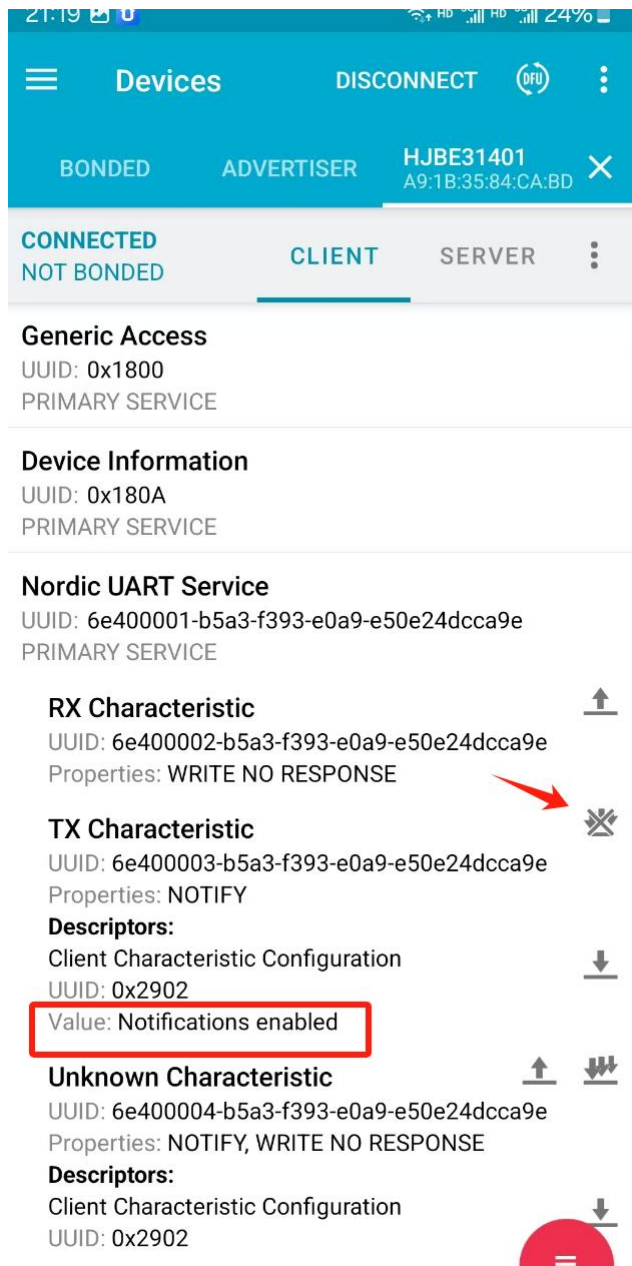
UUID: 0xFE59

PRIMARY SERVICE

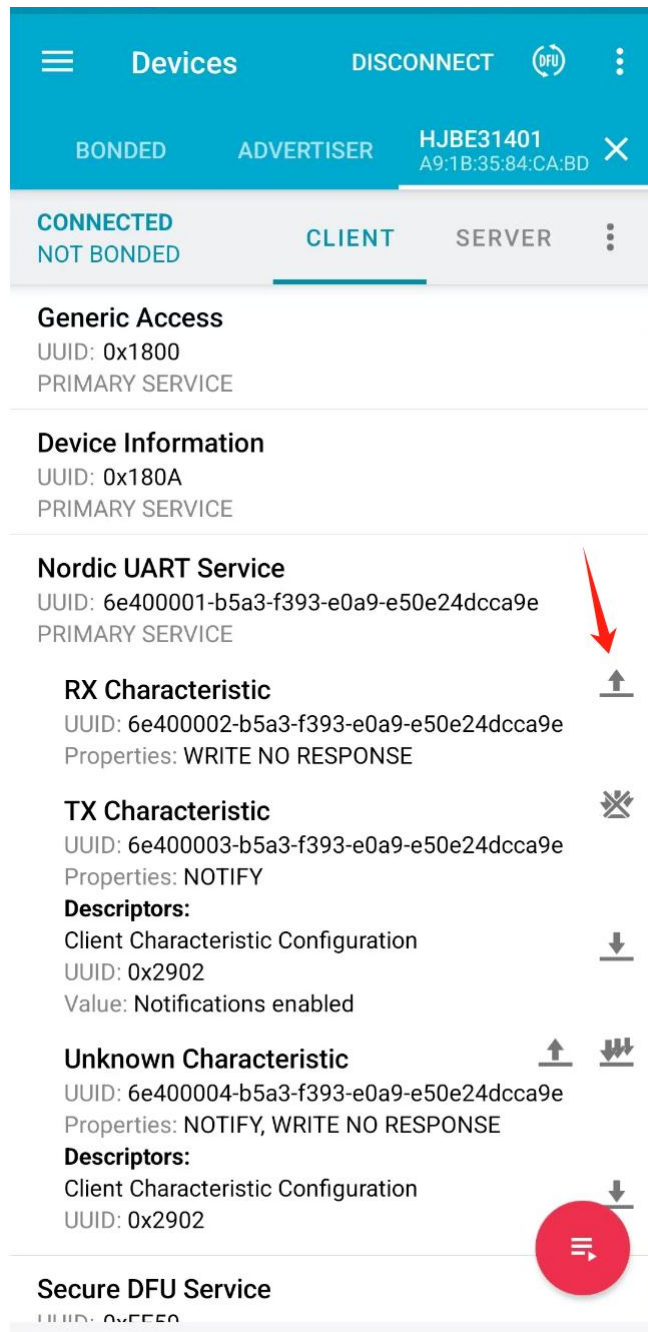
- 点击通知开关



- 看到图标变成这个形式，并且底部有通知已经使能提示

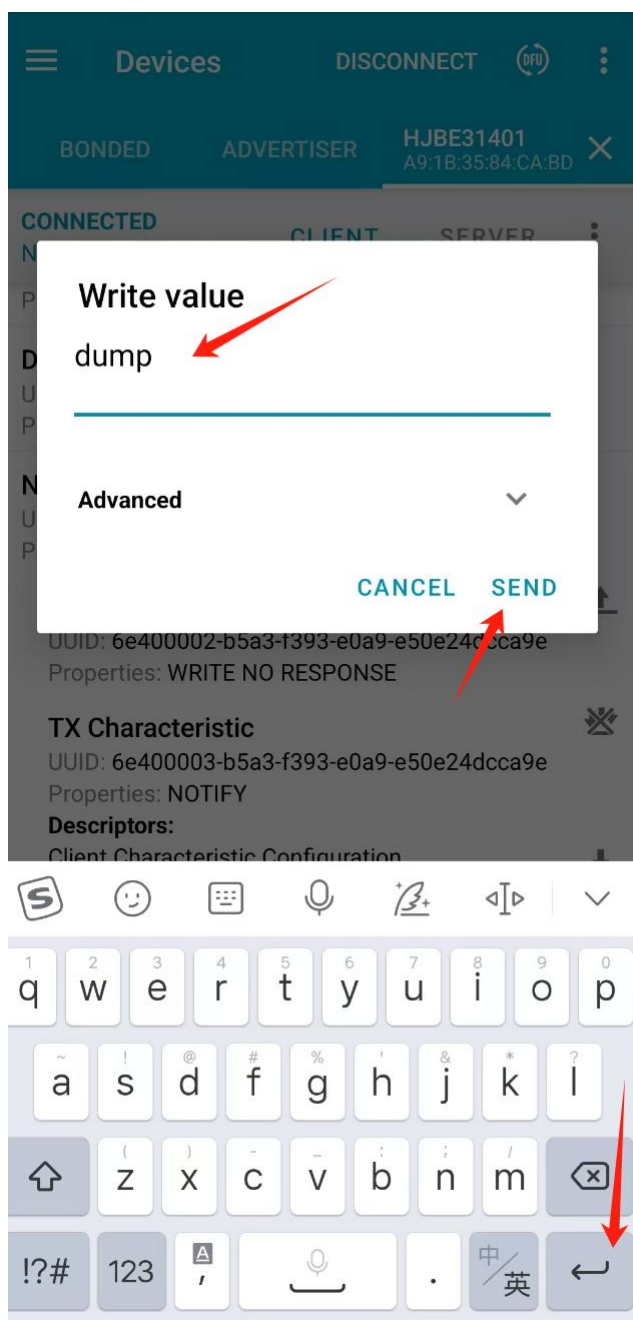


- 点击图示图标开始发送命令给传感器



- 输入传感器命令

先输入 dump, 然后按回车键, 最后按 SEND



- 看到传感器正常返回，说明已经跟传感器正常通讯。

The screenshot displays a Bluetooth connection interface. At the top, there's a status bar with 'CONNECTED' and 'NOT BONDED' labels, and tabs for 'CLIENT' and 'SERVER'. Below this, the 'PRIMARY SERVICE' is listed. The 'Device Information' section shows the UUID as 0x180A and the service as PRIMARY SERVICE. The 'Nordic UART Service' section shows the UUID as 6e400001-b5a3-f393-e0a9-e50e24dcca9e and the service as PRIMARY SERVICE. Under this service, there are two characteristics: 'RX Characteristic' (UUID: 6e400002-b5a3-f393-e0a9-e50e24dcca9e, Properties: WRITE NO RESPONSE, Value: dump) and 'TX Characteristic' (UUID: 6e400003-b5a3-f393-e0a9-e50e24dcca9e, Properties: NOTIFY). The 'TX Characteristic' value field is highlighted with a red box and contains the following data: Product Code: VIBRA-BLUE, Sensor type: 165, ID: 8, SW version: 1.0, HW version: 1.0, MAC: A9:1B:35:84:CA:BD, Range: 16, Frequency: 4000.00, and Ratio: 4790.2832. Below the characteristics, there's a 'Descriptors' section for 'Client Characteristic Configuration' (UUID: 0x2902, Value: Notifications enabled). At the bottom, there's an 'Unknown Characteristic' section. A red arrow points from the 'TX Characteristic' value field to the 'Value: Product Code: VIBRA-BLUE' line.

CONNECTED
NOT BONDED

CLIENT **SERVER**

PRIMARY SERVICE

Device Information
UUID: 0x180A
PRIMARY SERVICE

Nordic UART Service
UUID: 6e400001-b5a3-f393-e0a9-e50e24dcca9e
PRIMARY SERVICE

RX Characteristic
UUID: 6e400002-b5a3-f393-e0a9-e50e24dcca9e
Properties: WRITE NO RESPONSE
Value: dump

TX Characteristic
UUID: 6e400003-b5a3-f393-e0a9-e50e24dcca9e
Properties: NOTIFY
Value: Product Code: VIBRA-BLUE
Sensor type: 165
ID: 8
SW version: 1.0
HW version: 1.0
MAC: A9:1B:35:84:CA:BD
Range: 16
Frequency: 4000.00
Ratio: 4790.2832

Descriptors:
Client Characteristic Configuration
UUID: 0x2902
Value: Notifications enabled

Unknown Characteristic

传感器支持命令说明

命令需要以\n 或者\r 结束，即 ASCII 码 0A 或者 0D 结束。

● dump

获得传感器相关信息

```
<- dump
-> Product Code:VIBRA-BLUE
Sensor type: 165
ID:8
SW version:1.0
HW version:1.0
MAC:A9:1B:35:84:CA:BD
Range:16
Frequency:4000.00
Ratio:488.2813
```

分别为:

产品名称

编号

软件版本

硬件版本

MAC 地址

量程 图示 16g

采样频率 图示 4000Hz

加速度数值转为 ug 系数，即波形数据乘以该数得到 ug 的单位。在 start 命令获得的值要转为 ug 时候即乘以该数值。

● rms

获得振动特征值命令

一般用法为 rms 2048

```
<- rms 2048
->
{"name":"HJBE31401","MAC":"A9:1B:35:84:CA:BD","data":[{"type":165,"ts":1569888313,"pk_num":1,"sts":4,"bat_v":3330,"temp":19.938,"ax":0.0000,"ay":0.0000,"az":0.0000,"vx":0.0000,"vy":0.0000,"vz":0.0000,"dx":0.0000,"dy":0.0000,"dz":0.0000,"avx":1.0584,"avy":1.9239,"avz":-9.3867,"fx":0.0000,"fy":0.0000,"fz":0.0000,"fvx":0.0000,"fvy":0.0000,"fvz":0.0000}]}
```

name 蓝牙名称

MAC mac 地址

type 固定为 165，表示振动传感器

ts 保留不用

pk_num 数据包序号，每次上传累加
sts 4 代表传感器唤醒
bat_v 电池电压，单位 mv
temp 温度， °C
ax: x 轴加速度特征值 单位 m/s^2
ay: y 轴加速度特征值 单位 m/s^2
az: z 轴加速度特征值 单位 m/s^2
vx: x 方向速度量（振动值） 单位 mm/s
vy: y 方向速度量（振动值） 单位 mm/s
vz: z 方向速度量（振动值） 单位 mm/s
dx: x 方向位移 单位 um
dy: y 方向位移 单位 um
dz: z 方向位移 单位 um
avx: x 方向重力分量 单位 m/s^2
avy: y 方向重力分量 单位 m/s^2
avz: z 方向重力分量 单位 m/s^2
fx: x 方向振动频率 单位 Hz
fy: y 方向振动频率 单位 Hz
fz: z 方向振动频率 单位 Hz
fvx: x 方向加速度 FFT 之后的最大幅值 单位 m/s^2
fvy: y 方向加速度 FFT 之后的最大幅值 单位 m/s^2
fvz: z 方向加速度 FFT 之后的最大幅值 单位 m/s^2

注意：当除了三轴重力方向分量不为 0，其他振动量均为 0 的时候表示传感器处于静止状态

● 波形采集命令 start samples format

samples 为采集的采样点数即一组 x, y, z 的数量
samples 取值需要取 128 的整数倍，例如 1024, 4096 等
format 支持 1, 2, 3 个选项

1. 格式 1， 数据分别以 x, y, z 排列。每一行代表一个采样

```
<- start 128 1  
-> 493,-26,-1947  
491,-27,-1945  
493,-31,-1950  
495,-31,-1949  
496,-23,-1955  
495,-23,-1961  
501,-19,-1961  
500,-31,-1954  
498,-32,-1956
```

497,-29,-1953

496,-29,-1952

498,-27,-1951

2. format 2 以 json 格式传输

<- start 128 2

->

```
{"MAC":"A9:1B:35:84:CA:BD","type":"89","num":"1","sensor_id":"8","sum":"-188975","data":[50
5,-27,-1957,507,-32,-1950,499,-41,-1954,497,-38,-1952,500,-29,-1947,498,-33,-1946,501,-37,-19
54,505,-37,-1955,496,-39,-1950,500,-35,-1953,501,-35,-1956,499,-35,-1955,502,-36,-1952,499,-4
0,-1948,500,-32,-1946,497,-32,-1944,503,-39,-1950,508,-39,-1944,494,-31,-1946,498,-36,-1952,5
05,-38,-1948,502,-39,-1947,498,-28,-1946,500,-34,-1942,503,-34,-1945,502,-38,-1954,499,-41,-1
948,498,-31,-1943,504,-34,-1951,497,-37,-1944,501,-31,-1943,502,-24,-1940,502,-24,-1946,504,-
26,-1946,502,-32,-1948,502,-35,-1942,502,-33,-1938,496,-30,-1941,502,-35,-1937,503,-38,-1932,
501,-37,-1942,499,-35,-1950,493,-31,-1942,497,-33,-1940,495,-34,-1941,497,-31,-1944,498,-29,-
1943,499,-31,-1941,502,-34,-1946,496,-33,-1941,502,-28,-1942,496,-36,-1943,494,-33,-1943,501,
-32,-1934,496,-40,-1939,497,-40,-1945,502,-32,-1942,499,-31,-1943,499,-32,-1937,506,-35,-1939,
499,-34,-1943,490,-36,-1940,490,-33,-1946,492,-35,-1942,502,-37,-1944,503,-33,-1940,503,-30,-
1935,499,-28,-1941,502,-27,-1951,504,-24,-1950,497,-25,-1947,502,-33,-1942,508,-38,-1940,502,
-34,-1940,496,-31,-1943,497,-34,-1938,504,-31,-1945,503,-31,-1941,496,-31,-1943,499,-37,-1949,
498,-29,-1945,495,-29,-1940,493,-28,-1943,494,-38,-1936,499,-40,-1943,497,-36,-1941,498,-39,-
1940,498,-27,-1939,497,-32,-1940,499,-33,-1945,503,-34,-1944,500,-28,-1940,501,-31,-1935,502,
-30,-1939,499,-38,-1936,498,-30,-1944,492,-25,-1941,496,-32,-1944,491,-37,-1948,493,-37,-1935,
500,-34,-1936,497,-29,-1935,494,-29,-1936,497,-33,-1933,500,-35,-1935,502,-32,-1936,496,-29,-
1939,492,-33,-1934,494,-29,-1931,502,-33,-1935,504,-32,-1940,500,-34,-1934,501,-32,-1938,496,
-37,-1937,502,-31,-1935,496,-30,-1943,493,-36,-1950,496,-28,-1944,495,-25,-1942,503,-25,-1940,
505,-28,-1945,499,-35,-1939,493,-35,-1936,496,-30,-1939,503,-34,-1943,502,-33,-1935,504,-34,-
1931,499,-23,-1936]]}
```

{"MAC":"A9:1B:35:84:CA:BD", -- MAC 地址

"type":"89", 数据类型，固定为 89

"num":"1", 数据序列包，每个包加一

"sensor_id":"8", 传感器编号

"sum":"-188975", data 字段内所有数据累加和

"data": 传感器数据，分别以 x,y,z 排列，每个包为 128 组采样数据，即 128 个 x,y,z

3. format 3 以二进制格式。这种方式数据量最小

举例：start 4096 3

一个数据包定义如下

```
#define SAMPLES_PER_ACC_PACKET 128
```

```
struct ACC_PACKET
```

```
{
```

```
    uint32_t header_tag;    //固定头 0xfefefefe
```

```
    uint32_t pk;            //序列号，每个包加一
```

```

uint32_t size;           xyz 的数据量
int16_t xyz[SAMPLES_PER_ACC_PACKET*3]; //128*3
uint32_t crc;           //从 header_tag 到 xyz 结束的计算的 crc 校验码
uint32_t tail_tag;      //固定数据包尾 0x12121212
};

```

CRC 计算方法:

```

static const unsigned int CRC32_Table[256] =
{
    0x00000000, 0x77073096, 0xEE0E612C, 0x990951BA,
    0x076DC419, 0x706AF48F, 0xE963A535, 0x9E6495A3,
    0x0EDB8832, 0x79DCB8A4, 0xE0D5E91E, 0x97D2D988,
    0x09B64C2B, 0x7EB17CBD, 0xE7B82D07, 0x90BF1D91,
    0x1DB71064, 0x6AB020F2, 0xF3B97148, 0x84BE41DE,
    0x1ADAD47D, 0x6DDDE4EB, 0xF4D4B551, 0x83D385C7,
    0x136C9856, 0x646BA8C0, 0xFD62F97A, 0x8A65C9EC,
    0x14015C4F, 0x63066CD9, 0xFA0F3D63, 0x8D080DF5,
    0x3B6E20C8, 0x4C69105E, 0xD56041E4, 0xA2677172,
    0x3C03E4D1, 0x4B04D447, 0xD20D85FD, 0xA50AB56B,
    0x35B5A8FA, 0x42B2986C, 0xDBBBC9D6, 0xACBCF940,
    0x32D86CE3, 0x45DF5C75, 0xDCD60DCF, 0xABD13D59,
    0x26D930AC, 0x51DE003A, 0xC8D75180, 0xBF06116,
    0x21B4F4B5, 0x56B3C423, 0xCFBA9599, 0xB8BDA50F,
    0x2802B89E, 0x5F058808, 0xC60CD9B2, 0xB10BE924,
    0x2F6F7C87, 0x58684C11, 0xC1611DAB, 0xB6662D3D,
    0x76DC4190, 0x01DB7106, 0x98D220BC, 0xEFD5102A,
    0x71B18589, 0x06B6B51F, 0x9FBE4A5, 0xE8B8D433,
    0x7807C9A2, 0x0F00F934, 0x9609A88E, 0xE10E9818,
    0x7F6A0DBB, 0x086D3D2D, 0x91646C97, 0xE6635C01,
    0x6B6B51F4, 0x1C6C6162, 0x856530D8, 0xF262004E,
    0x6C0695ED, 0x1B01A57B, 0x8208F4C1, 0xF50FC457,
    0x65B0D9C6, 0x12B7E950, 0x8BBEB8EA, 0xFCB9887C,
    0x62DD1DDF, 0x15DA2D49, 0x8CD37CF3, 0xFBD44C65,
    0x4DB26158, 0x3AB551CE, 0xA3BC0074, 0xD4BB30E2,
    0x4ADFA541, 0x3DD895D7, 0xA4D1C46D, 0xD3D6F4FB,
    0x4369E96A, 0x346ED9FC, 0xAD678846, 0xDA60B8D0,
    0x44042D73, 0x33031DE5, 0xAA0A4C5F, 0xDD0D7CC9,
    0x5005713C, 0x270241AA, 0xBE0B1010, 0xC90C2086,
    0x5768B525, 0x206F85B3, 0xB966D409, 0xCE61E49F,
    0x5EDEF90E, 0x29D9C998, 0xB0D09822, 0xC7D7A8B4,
    0x59B33D17, 0x2EB40D81, 0xB7BD5C3B, 0xC0BA6CAD,
    0xEDB88320, 0x9ABFB3B6, 0x03B6E20C, 0x74B1D29A,
    0xEAD54739, 0x9DD277AF, 0x04DB2615, 0x73DC1683,

```

```

0xE3630B12, 0x94643B84, 0x0D6D6A3E, 0x7A6A5AA8,
0xE40ECF0B, 0x9309FF9D, 0x0A00AE27, 0x7D079EB1,
0xF00F9344, 0x8708A3D2, 0x1E01F268, 0x6906C2FE,
0xF762575D, 0x806567CB, 0x196C3671, 0x6E6B06E7,
0xFED41B76, 0x89D32BE0, 0x10DA7A5A, 0x67DD4ACC,
0xF9B9DF6F, 0x8EBEEFF9, 0x17B7BE43, 0x60B08ED5,
0xD6D6A3E8, 0xA1D1937E, 0x38D8C2C4, 0x4FDFF252,
0xD1BB67F1, 0xA6BC5767, 0x3FB506DD, 0x48B2364B,
0xD80D2BDA, 0xAF0A1B4C, 0x36034AF6, 0x41047A60,
0xDF60EFC3, 0xA867DF55, 0x316E8EEF, 0x4669BE79,
0xCB61B38C, 0xBC66831A, 0x256FD2A0, 0x5268E236,
0xCC0C7795, 0xBB0B4703, 0x220216B9, 0x5505262F,
0xC5BA3BBE, 0xB2BD0B28, 0x2BB45A92, 0x5CB36A04,
0xC2D7FFA7, 0xB5D0CF31, 0x2CD99E8B, 0x5BDEAE1D,
0x9B64C2B0, 0xEC63F226, 0x756AA39C, 0x026D930A,
0x9C0906A9, 0xEB0E363F, 0x72076785, 0x05005713,
0x95BF4A82, 0xE2B87A14, 0x7BB12BAE, 0x0CB61B38,
0x92D28E9B, 0xE5D5BE0D, 0x7CDEFB7, 0x0BDBDF21,
0x86D3D2D4, 0xF1D4E242, 0x68DDB3F8, 0x1FDA836E,
0x81BE16CD, 0xF6B9265B, 0x6FB077E1, 0x18B74777,
0x88085AE6, 0xFF0F6A70, 0x66063BCA, 0x11010B5C,
0x8F659EFF, 0xF862AE69, 0x616BFFD3, 0x166CCF45,
0xA00AE278, 0xD70DD2EE, 0x4E048354, 0x3903B3C2,
0xA7672661, 0xD06016F7, 0x4969474D, 0x3E6E77DB,
0xAED16A4A, 0xD9D65ADC, 0x40DF0B66, 0x37D83BF0,
0xA9BCAE53, 0xDEBB9EC5, 0x47B2CF7F, 0x30B5FFE9,
0xBDBDF21C, 0xCABAC28A, 0x53B39330, 0x24B4A3A6,
0xBAD03605, 0xCDD70693, 0x54DE5729, 0x23D967BF,
0xB3667A2E, 0xC4614AB8, 0x5D681B02, 0x2A6F2B94,
0xB40BBE37, 0xC30C8EA1, 0x5A05DF1B, 0x2D02EF8D
};

/**
 * [_getCRC32 使用查表法计算 CRC32]
 * @param buffer [缓存地址]
 * @param bufferLen [缓存长度]
 * @return [计算出的 CRC32 值]
 */
unsigned int _getCRC32(uint32_t *flash_addr, unsigned int bufferLen)
{
    unsigned int crc32Value = 0xffffffff;

    unsigned char *pTmpBuffer = (uint8_t*)flash_addr;

```

```
while(bufferLen--)\n{\n    crc32Value = CRC32_Table[(crc32Value^ *pTmpBuffer++) & 0xff] ^ (crc32Value >> 8);\n}\n\nreturn (crc32Value^0xffffffff);\n}
```

- **timeout 命令**

默认情况下，传感器如果在 60 秒内没有与主机有数据通讯的话会自动断开链接并进入休眠状态。可以通过 **timeout** 命令修改这个值或者关闭这个自动断开休眠功能（即传感器只要有蓝牙链接就不会自动休眠）。

关闭超时自动休眠功能： **timeout 0**

修改超时时长为 2 分钟： **timeout 120**

- **RESET 重启传感器**

在某些情况下，可能需要重新启动传感器