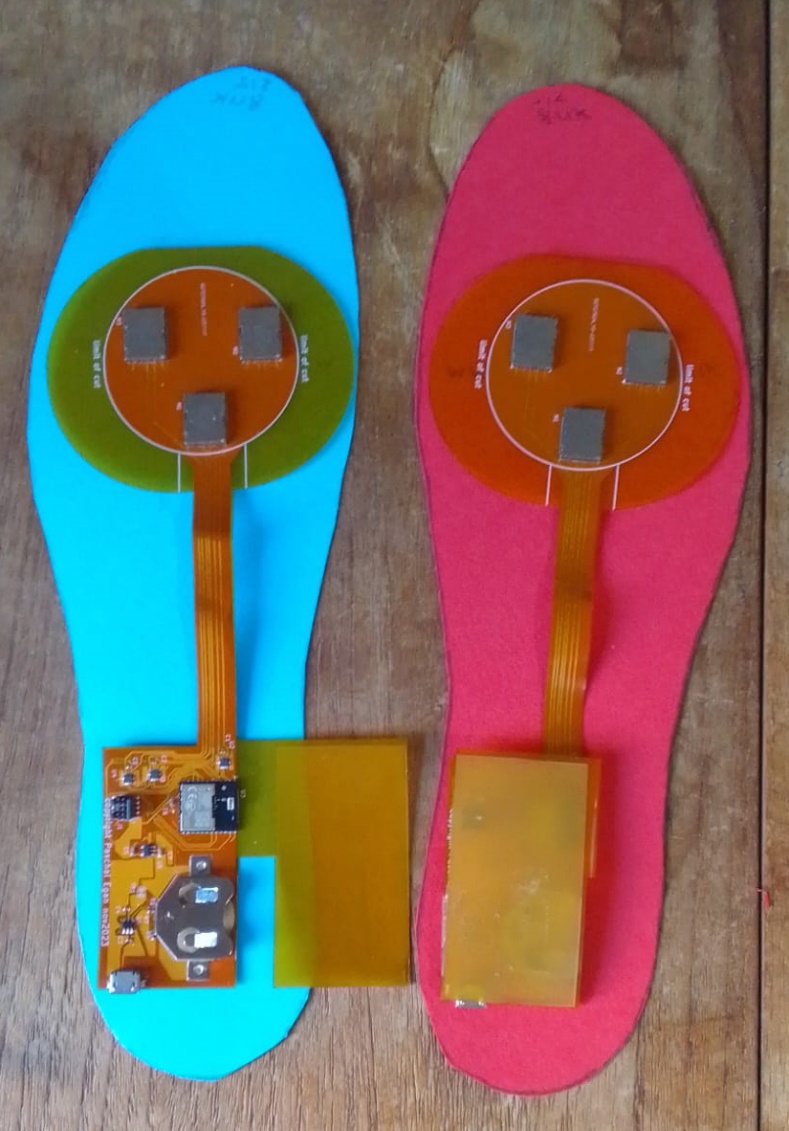
[WWW.NOISEPOWER.CO.UK](http://WWW.NOISEPOWER.CO.UK)

USER MANUAL

NoisePower Vib1

1

Contents

[Out of Box: 3](#_Toc159922742)

[Getting started: 3](#_Toc159922743)

[Testing 3](#_Toc159922744)

[Assembly 4](#_Toc159922745)

[Phone App 4](#_Toc159922746)

[Summary of functionality of App 4](#_Toc159922747)

[Inside App 5](#_Toc159922748)

[Partition of code: 5](#_Toc159922749)

[Flexi PCB App 6](#_Toc159922750)

[PCB Hardware 7](#_Toc159922751)

[Appendix A: Android Device code: 8](#_Toc159922752)

[Designer view 8](#_Toc159922753)

[Scanning 8](#_Toc159922754)

[Connecting: 8](#_Toc159922755)

[When Connected 9](#_Toc159922756)

[Detecting Walking: 9](#_Toc159922757)

[Timeouts for turning motors on: 10](#_Toc159922758)

[Manual Control: 10](#_Toc159922759)

[Housekeeping 11](#_Toc159922760)

[Variables 11](#_Toc159922761)

[Disconnecting 11](#_Toc159922762)

Introduction

This is the user manual/ instructions for the “NoisePower Vib1”. This is a development system for the study of the therapeutic use of vibrations. The target use is to develop protocols for an aid for people with Gait Freeze (typically due to Parkinsons disease).

The system depends on an android device with an IMU inertia measuring unit) and a Bluetooth Low Energy connection. (not supplied- typically an android smart phone).

The Vib1 is a system for applying vibrations to parts of the human body, with therapeutic intent. The design allows researchers, and others, to investigate the benefits of vibrations at specific times, in relation to a subject walking. As delivered, it produces a 5second vibration to the subject feet every time that they have been walking and then stop. If walking starts again within the 5s the vibrations will cease, else they will cease after 5s. There will be no more vibrations until walking is resumed, and again vibrations will occur on cessation. The device is intended as a **research tool- not a finished product for an end user.**

The system will be supplied preprogrammed. Programming skills are not required to use (but will be to make changes to functionality). This is a chargeable service Noisepower can provide- or there is sufficient information for someone with basic coding skills to do alterations).

The code in the flexi PCB is written in the C programming language, using the Simplicity Studio Integrated design environment. The code for the android device is offered as an executable (hence no programming skills needed) and as source code (though NOT open source). It is written in MIT APPinventor. Other development paths are available (e.g. a demo from silicon vendor written in Kotlin/ Java).

# Out of Box:

There will be two flexi PCBs, one marked L and the other R. There is also a USB pens drive with this user manual on, and more technical information, including the code. In addition, there are supplied two sheets of Conford, card with left and right insole images printed on them. If a pair are bought a programming cable is also supplied (can be bought separately).

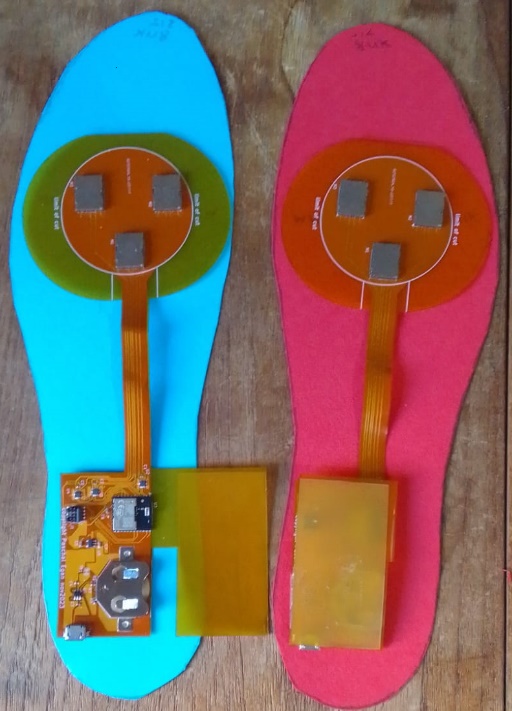
# Getting started:

Make sure both flex PCB are fully charged (suggest at least 4 hours connected to a standard USB charger).

## Testing

Install “Lightblue” (or equivalent) app on the android device. Should detect Blinky L and Blinky R. Connect to one of them and send a HEX number 01 to the service 5B026510-4088-C297-46D8-BE6C736A087A. The motors on that device will vibrate. Repeat for the other side. (Same service ID- different device name).

## Assembly

 A pair of colorful insoles

Description automatically generated

Check the batteries are fitted (supplied fitted in most areas- some countries shipping rules require them to be sourced locally)

* Tape across the FR4 cover (See over)
* Cut the insole to shoe size.
* Tape (Sellotape or glue stick) the PCB to the insole.
* Cover the rear heal section with an insert.

Assembly will vary with experiment intent. The supplied two insoles are printed on Canford card. Experience suggests, to get good transmission into the foot, stiffness is required. Other insoles can be used.

If the user has tight fitting shoes – it is worthwhile to lightly cover the vibration motors with a thin film, or laminate with a thin layer. Sellotape is a possibility- to protect motors while moving foot for fit.

# Phone App

To illustrate the operation of the vibration system, as a complete system, there is a phone app for Android devices. This is available as a downloadable executable. The editable code is also available at [www.noisepower.co.uk](http://www.noisepower.co.uk) note: This is NOT open source and only licensed for use with Noisepower devices.

## Summary of functionality of App

Assuming the user has within vicinity two powered flex PCB units- one which offers a name Blink L and the other Blinky R, the app on the device can connect to each in term:

* Scan L (and select Blink L from list)
* Connect L (status should go to connected L)
* Scan R (and select blink R from list)
* Connect R (status should go to both connected)

A screenshot of a computer

Description automatically generated

To test the motors, you can press LedOn and LedOff buttons. The motors should vibrate on (and off),

If you start to walk- the status will go to walking. When you stop, the status should go to “Stationary”. The front pocket of a pair of trousers is a possible location for the device.

## Inside App

The connect/ disconnect is standard for Bluetooth apps. The only difference to normal connection of a sensor to a BLE device is that there are two devices The function for the walking calls up the pedometer. This interrogates Inertia Measurement Unit (IMU) sensors. An android device might have a dedicated microprocessor to manage the IMU unit and detect walking and non-walking and steps. It can be read and will return how many steps since turned on, and whether currently walking etc. On detecting both Bluetooth units this process started. A timer is set to every 1.5second. On the timer event the step count is updated. If it has not increased, the user is deemed to have stopped walking. The status is changed, and the motors started.

The motor control is a simple write of a 0x00 (motor off) or 0x01 (motor on) to each of the connected BLE services.

When the motors are on- every 1s the pedometer is interrogated to see if walking now restarted, and after 5 seconds the motors are switched off independent of pedometer state. They will not activate again until walking detected (And then stopped).

## Partition of code:

If the system is to be adapted, partition decisions will need to be made. More than 0/1 could be sent from the mobile app. Possibilities include sending alternately to left or right feet, or slowly increasing the intensity as time not walking progresses. The device end would need to be altered to match.

In addition, code could be developed to read the IMU sensors directly. This might facilitate reading the IMU directly. Possibilities include measuring the spectral content (Fourier Transform) to discriminate when gait is frozen (ie higher frequency) or a digital phase lock loop to detect when lock lost.

At the device code end, it possible patterns could be introduced. These might rotate for example. So a 1 would trigger different actions, and possibly 2, 3, ….. all have different functionality.

# Flexi PCB App

The flexi PCB contains a BG220p Bluetooth module. If required, this can be programmed with a Jlink (from segger) programmer or a thunderbird board (from silicon image).

A close up of a circuit board

Description automatically generated

The code is developed using C language with silicon lab’s Simplicity Studio. The starting code is SoC Blinky. This connects with code on a mobile device and drives a port pin high and low dependant on whether a “0” or “1” received. Silicon Labs have a mobile app called EFRconnect, and a small demo board with an led on that port pin. This has been recompiled to drive a 1 or a zero to 5 port pins.

The changed code:

*if (data\_recv == 0x00) {*

*GPIO\_PinOutClear(gpioPortB, 2);*

*GPIO\_PinOutClear(gpioPortB, 3);*

*GPIO\_PinOutClear(gpioPortB, 4);*

*GPIO\_PinOutClear(gpioPortB, 0);*

*GPIO\_PinOutClear(gpioPortB, 1);*

*GPIO\_PinOutClear(gpioPortA, 8);*

*app\_log\_info("LED off.\n");*

*} else if (data\_recv == 0x01) {*

*GPIO\_PinOutSet(gpioPortB, 2);*

*GPIO\_PinOutSet(gpioPortB, 3);*

*GPIO\_PinOutSet(gpioPortB, 4);*

*GPIO\_PinOutSet(gpioPortB, 0);*

*GPIO\_PinOutSet(gpioPortB, 1);*

*GPIO\_PinOutSet(gpioPortA, 8);*

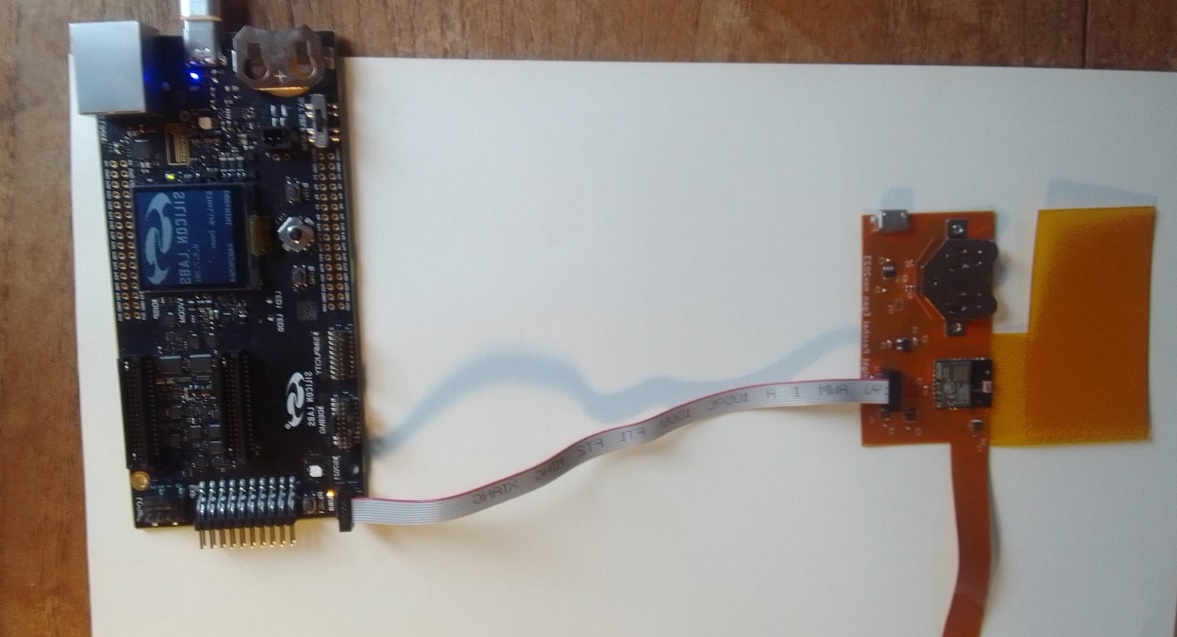
# PCB Hardware

The flexi PCB contains as its core a BGM220p Bluetooth module. The system is powered by a 3.7V Li/Ion cell. There is a USB socket. This is to recharge the cell- not for programming or data communication.

The three ports pins from the ARM go to specialist LRA motor controllers. These detect the resonant frequency of each motor. There is a common port pin, taken to the enable pin of each controller. Untested functionality is that this pin is PWM to give an analogue of the intensity of vibration. Current functionality is as a general on/off.

Each controller powers one LRA motor. These have a resonant frequency of approximately 172Hz.

Reprogramming can be achieved by a Jlink adaptor (from megger) such as on the thunderbird board to the 10 pin mini simplicity. Note: A reversed cable needs to be used. One will be supplied with orders of two or more ver1.0 boards. **DO NOT PLUG IN A STANDARD 0.05” 10 way cable.** A standard cable can be converted by removing carefully with a craft knife the polarising bump at one end and fitting so that pin one is at the motors end of the socket. See below:



Note position of pin1- red cable.

# Appendix A: Android Device code:

The full code can be obtained from [www.noisepower.co.uk/downloads](http://www.noisepower.co.uk/downloads)

The blocks are described here:

## Designer view

A screenshot of a cell phone

Description automatically generated

## Scanning

A screenshot of a computer

Description automatically generated



The right scan is the same.

## Connecting:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## When Connected

A screenshot of a computer

Description automatically generated

Timers and pedometer started. (could move to pedometer always on if wished to investigate detection / drop off from walking independently of the motors).

A screenshot of a computer

Description automatically generated

Updating labels when connected.

## Detecting Walking:

A screenshot of a computer

Description automatically generated

Every time a step detected the outputs driven to “0” (turn off motors).

## Timeouts for turning motors on:

A screenshot of a computer program

Description automatically generated

## Manual Control:

A screenshot of a computer program

Description automatically generated

The manual “Off” is same but sends a “0” and reads other button.

## Housekeeping

### Variables







### Disconnecting

A screenshot of a computer

Description automatically generated

A close up of a button

Description automatically generated

A screenshot of a computer

Description automatically generated

## 