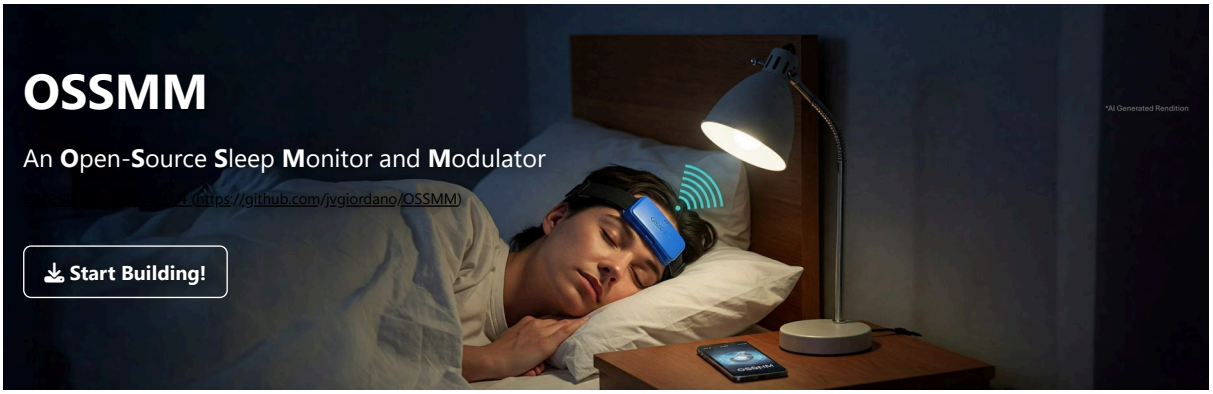
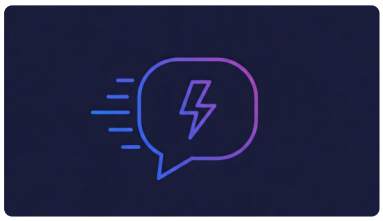


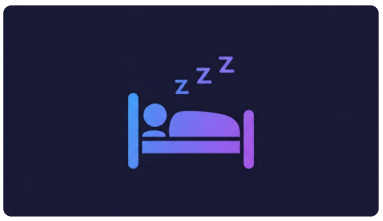
OSSMM breaks down the cost barriers in sleep research by providing researchers and enthusiasts with an affordable, accurate, and fully open-source platform for both sleep monitoring and modulation experiments. Build your own system for under €60.



Quick Intro

New to OSSMM? Get a rapid overview of what it is, how it works, and why it matters for sleep research.

Get Started



Complete Build Guide

Deep dive into OSSMM's capabilities, construction process, and cost breakdown. Everything you need to build your own system.

Start Building

Sleep Staging Performance

Currently under Assessment for 4-Stage Sleep Classification

Learn more

Current Version: V1.0.4

Quick Introduction (https://jvgiordano.github.io/OSSMM/quick-intro/)

5 minute read

Welcome to OSSMM!

OSSMM (Open-Source Sleep Monitor and Modulator) is the world's first open-source hardware and software system designed for both sleep monitoring and modulation.

Who is it for?

OSSMM is for researchers and sleep enthusiasts in need of an affordable and accurate sleep monitoring system. It's also for those who need a platform which can implement automated sleep modulation.

Building your own OSSMM requires some basic-to-moderate electronics and 3D printing knowledge. We provide resources for learning everything you need to know, and a detailed Build Guide for assembling your own.

This is a great starter project for electronics and 3D printing. For first time builders with basic background knowledge, assembly should take 4-5 hours for the first unit. Those who are experienced can easily assemble a unit in under 2 hours.

Who made it?

The OSSMM project was developed as part of Jonny Giordano's thesis at the Hamilton Institute at Maynooth University.

The Goal:

OSSMM was created to address the prohibitive entry cost into quality sleep research. By providing researchers and sleep enthusiasts with an affordable platform that can be assembled locally, this accessible system promotes:

- Cost-effective large-scale, long-term sleep studies
- Data collection in participants' natural home environments
- Effective monitoring for under €40 per unit (as of 12/2024)[1]

Most importantly, OSSMM enables sleep modulation experiments — filling a gap where no comparable off-the-shelf system currently exists.

OSSMM Headband on display with strap fully stretched out.

Macro photo of OSSMM Headband. USB-C, LED port, ON/Reset switch, and microphone port are visible. €1 euro coin for size reference.

What are “sleep monitoring” and “sleep modulation”?

Sleep monitoring is the recording of physiological signals to assess sleep duration, quality, and architecture. This typically involves sleep staging - the categorization of sleep-wake states. Five-stage systems include N1, N2, N3, REM (Rapid Eye Movement), and Wake states, while four-stage systems use Deep Sleep, Light Sleep, REM, and Wake.

Sleep modulation is the intentional use of non-invasive techniques to modify, regulate, or enhance sleep architecture and target specific sleep signatures. Goals include improving sleep quality, optimizing sleep stages, and enhancing associated cognitive and physiological functions. Common techniques include acoustic, optical, and electrical stimulation, and physical arousal.

Understanding the Build Process

Creating your own OSSMM system involves five main steps that follow our detailed build guides:

Step 1: 3D Printables - Print the receiver using TPU filament and the electronics case using PLA. Print times will depend on your printer and experience level.

Step 2: Electronics Assembly - Standard soldering to connect the electronic components together and install them in the 3D printed case. This should take no more than 4-6 hours for a beginner. Experienced OSSMM builders can assemble comfortably under 50 minutes.

Step 3: Major Parts Assembly - Combine the three principal components: headband, receiver, and electronic case. This should take no more than an hour.

Step 4: Software - Upload code to the microcontroller and install the Android app. This should take no more than an hour.

Step 5: Final Checks - Systematic testing to verify all sensors function correctly. This should take no more than 30 minutes.

The modular design means you can work on these steps independently or collaborate with others who have different expertise in 3D printing, electronics, or software. We estimate a first build to require ~2 Days, including over night print times.

Key Design Features:

- **Reusable silicone wet-dry electrodes** - no conductive gel needed
- **Quick-change parts** - easy repair and hygiene
- **Battery powered with no exposed wiring** - 15+ hour run-time
- **Fully open-source hardware designs and software code**
    - Complete access for researcher modifications
    - Full transparency of sleep staging algorithms and hardware code

Device Overview (V1.0.4)

OSSMM consists of a wearable headband that collects physiological data and transmits it wirelessly via Bluetooth Low Energy (BLE) to a dedicated Android application. Only Android is supported at this time.

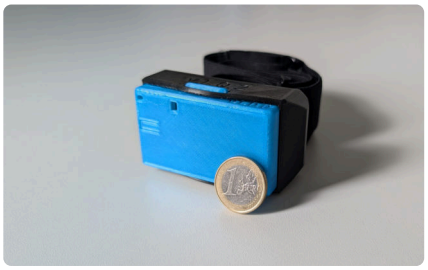


Fig 1: Front of the OSSMM headband compared with a €1 coin.

Fig 2: Back of the OSSMM headband, with the silicone electrodes and PPG sensor visible.

Benefits:

- **Minimal Design**: Just 11 components (excluding wires)
- **Comfortable Electrodes**: Elastic band with integrated silicone wet-dry electrodes (commercial heart rate monitor strap)
- **Standard Electronics**: Four readily available off-the-shelf circuit boards
- **Locally Manufacturable Housing**: 3D printable design
- **Convenient Power**: USB-C rechargeable battery for easy charging

Technical Specifications:

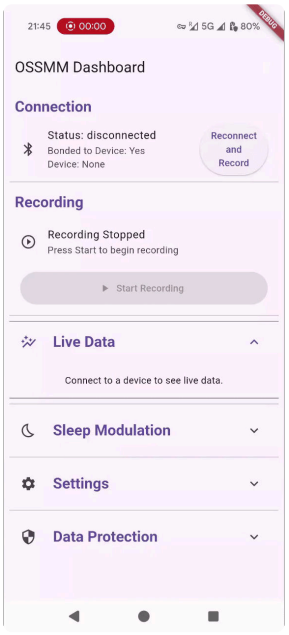
- **Dimensions: 79.1 x 45.2 x 30 mm** (3.12 x 1.78 x 1.18 in)
- **Weight: 76.5 grams** (or 2.7 ounces with a 150 mAh battery)
- **Sampling Frequency: +250 Hz** (500MB of data per 8 hour recording)
- **Battery: 120-220 mAh** (est. 15-27+ hour run time, sleep monitoring only)

Physiological Measurements:

- **Brain Activity:** Frontal Electroencephalography (EEG)
- **Eye Movement:** Electrooculography (EOG)
- **Head Movement:** Via onboard Inertial Measurement Unit (IMU)
- **Heart Rhythm:** Photoplethysmography (PPG)

Note: While the hardware includes a microphone for sound data collection, this feature is not activated in version 1.0.4.

Quick App Demo



Short Demo of OSSMM app. Real-time app usage is smoother than can be shown in this GIF.

Performance:

OSSMM promises to offer more accurate sleep staging than many commercial wearables (smart watches, rings) at a fraction of the price.

Note: OSSMM is currently under technical validation for 4-stage sleep classification.

Sleep Modulation Capabilities:

OSSMM V1.0.4 incorporates a commercial off-the-shelf vibration motor (similar to those in mobile phones) as a stimulus mechanism for sleep modification experiments. The vibration motor was chosen as the reference stimulus because:

1. It demonstrates the system's robust power handling (60+ mA during operation)
2. It proves the platform can easily accommodate other stimulus methods including speakers, LEDs, tDCS, and tACS

Future versions aim to analyze sleep data in near-real-time (within 30-60 seconds) to potentially trigger sleep modulation based on detected sleep stages.

Start Building Now

We recommend reviewing the Getting Started (https://jvgiordano.github.io/OSSMM/getting-started/) before beginning your OSSMM build. This page contains detailed information about OSSMM and all the pre-requisites.

To build your own OSSMM system, follow these pages in order:

1. 3D Printables (https://jvgiordano.github.io/OSSMM/printables) - Print out the printable components
2. Electronics Assembly Guide (https://jvgiordano.github.io/OSSMM/electronics-assembly/) - Combine the electronics and 3D printed case
3. Major Parts Assembly (https://jvgiordano.github.io/OSSMM/final-assembly/) - Assemble the 3 principal components: headband, receiver, electronic case
4. Software (https://jvgiordano.github.io/OSSMM/software/) - Upload OSSMM code to the MCU (Microcontroller), and install the OSSMM apk (app file) on your Android device
5. Final Checks & Completion (https://jvgiordano.github.io/OSSMM/final-checks) - Verify your OSSMM system collects data as intended

Additional Notes:

1. V1.0.4 requires an Android device with the dedicated companion app to work. While currently requiring an Android device, any platform supporting high-priority BLE transmission could potentially work (iPhone, Raspberry Pi, etc.). The cost estimate does not include the Android device. ↩

Updated: May 16, 2025

Getting Started - A Deep Dive Introduction (https://jvgiordano.github.io/OSSMM/getting-started/)

11 minute read

This guide provides an in depth introduction to the OSSMM system, explaining its functionality, capabilities, cost, and the prerequisites for building your own.

Please don't be intimidated by the Table of Contents (on any page)!

Great care has been taken to explain each step in pedantic detail, so what may appear as complexity is closer to excessive detail. Each step has been broken down into many small parts to eliminate any confusion. You'll likely find yourself breezing through them much faster than you'd expect. The extensive detail is there to guide you, not to overwhelm you.

How It All Works

Major System Parts

The OSSMM system consists of 4 parts:

1. **The User (participant)**
2. **The OSSMM Headband**
3. **Android Device and Dedicated App**
4. **You - The Researcher**

AI generated rendition of OSSMM System. 1) The User 2) OSSMM Headband 3) Android Device running OSSMM app

Users wear the OSSMM headband each night during sleep. Version 1.0.4 of the headband collects:

- Head movement
- Eye movement (EOG)
- Frontal brain activity (EEG)
- Heart Rhythm data (PPG)

This data is streamed via Bluetooth Low Energy (BLE) to the OSSMM companion app on an Android device, which saves the data on local storage for later analysis. The system can be used repeatedly (e.g. 20 nights) as long as the headband is charged, and there is enough storage on the Android device.

As a researcher, you can later collect the Android device and analyze the data as desired according to your study design.

Modularity

The OSSMM headband can be built locally with basic tools and basic electronics knowledge. It is possible to:

- Modify the hardware design using free, browser based CAD software (OnShape)
- Customize the Android app code or headband software
- Analyze the data with your own methods and algorithms

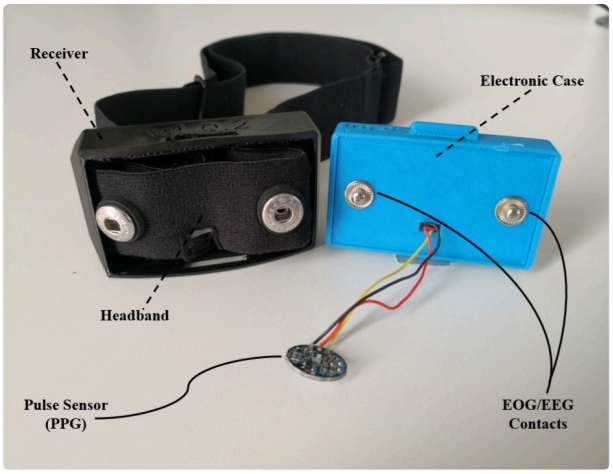
Future Possibilities

With future work it will be possible to have:

1. Near Real-Time Sleep Staging (on the Android device)
2. Automated Sleep Modulation Stimulus Delivery (based on sleep stage or custom signal detection)
3. App support beyond Android (iPhone, Raspberry Pi)
4. Audio Data Collection for additional analyses (e.g., sleep apnea detection)
5. Cloud Storage for remote collection and back-up (if desired)

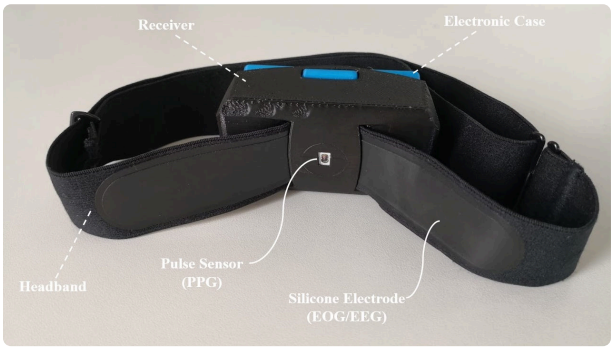
OSSMM Headband - Principal Components

The OSSMM Headband comprises 3 principal components: An electronic case, a headband, and a receiver.

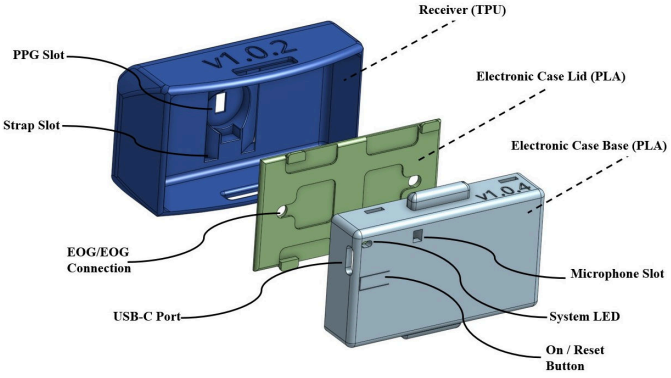


Open view of the three major components, all disconnected: Headband, Receiver, and Electronic Case.

The electronic case consists of two 3D printed parts which house the electronics. The headband is an adjustable strap with silicone electrodes for EOG/EEG signal collection. These two components connect through the 3D printed receiver made from soft filament.



Rear view of OSSMM headband.



Annotated Printables. TPE = Thermoplastic Elastomer, PLA = Polylactic Acid

OSSMM App and Data

The companion app serves multiple purposes:

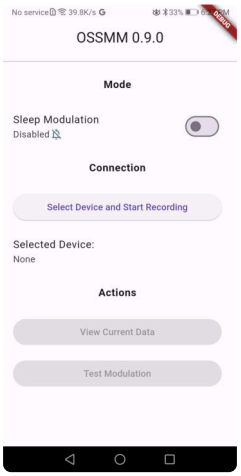
- Stores data collected by the headband to local storage
- Provides accurate date-time information
- Will support near real-time sleep staging in future versions

v0.9.0 App Features

Buttons, Toggles, and Connections

The app allows a scan of nearby Bluetooth devices with the “Select Device and Start Recording” button. Once the headband is selected, the app establishes a BLE connection with the headband, initiates data collection upon connection, and stores data continuously, even if the connection is temporarily lost.

The “Sleep Modulation” toggle enables BLE signals to be sent from the Android device to the headband. Once enabled, the user can press the “Test Modulation” button during an established connection to activate the headband’s vibration motor in a double-blink pattern.

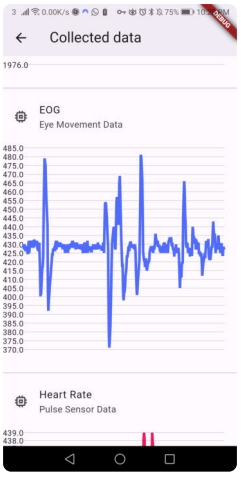


Main page of OSSMM app

Real-Time Data Visualization

Selecting “View Current Data” once a connection is established will open to a new page with live graphs of the accelerometer, gyroscope, eye movement (EOG), and pulse sensors (PPG) data.

While both eye movement and brain activity are captured in the same signal, only the EOG component can be reliably identified visually in real-time by the user. Along with the other signals like head movement and pulse, you can use eye movement to confirm that the headband is properly functioning.



Real Time Graph of EOG Data with several eye movements (spikes)

Data Collection

The app continuously saves data in CSV format. At a sampling rate of 250 Hz, approximately 500 MB of storage is required for an 8-hour sleep recording.

Each data sample collected by the headband and sent to the companion app includes ten columns:

Column	Description
Datetime	Precise timestamp when the sample was recorded by the app (down to millisecond)
transNum	Transmission number sequence (0-65,535) sent by the headband
eog	Combined EOG/EEG signal
hr	Heart rhythm/pulse data
accX, accY, accZ	Acceleration in three axes
gyroX, gyroY, gyroZ	Angular velocity in three axes

Note: The transmission number resets to 0 after reaching 65,535 (the maximum value for a 2-byte unsigned integer) and helps identify if BLE updates are being lost.

DateTime	transNum	eog	hr	accX	accY	accZ	gyroX	gyroY	gyroZ
11:11:49.186 PM	0	64	1	802	876	738	2004	1998	1998
11:11:49.191 PM	1	89	1	802	877	738	2004	1999	1997
11:11:49.192 PM	2	2	0	802	877	738	2004	1999	1997
11:11:49.192 PM	3	0	0	799	877	737	2004	1999	1997
11:11:49.192 PM	4	2	1	799	877	737	2004	1999	1997
11:11:49.192 PM	5	0	1	798	877	736	2004	1997	1996
11:11:49.192 PM	6	1	702	798	877	736	2004	1993	1995
11:11:49.192 PM	7	1	699	800	876	736	2004	1993	1995
11:11:49.192 PM	8	0	702	802	875	738	2004	1994	1994
11:11:49.197 PM	9	0	701	802	875	738	2004	1994	1994
11:11:49.197 PM	10	1	701	806	874	738	2005	1995	1992
11:11:49.197 PM	11	0	700	806	874	738	2005	1999	1993
11:11:49.197 PM	12	0	700	808	874	739	2007	1999	1993
11:11:49.198 PM	13	0	702	808	874	739	2007	2005	1993
11:11:49.198 PM	14	0	700	807	875	739	2007	2005	1993
11:11:49.198 PM	15	1	700	804	874	738	2008	2009	1994

Portion of a sample CSV file

Advanced note: Accelerometer and Gyroscope values have been shifted so that only unsigned integers are used during BLE transmission. This ensures high speed BLE transmission rates by decreasing the memory required for each update.

Build Your Own - What You Need to Know

The purpose of this section is to inform on the required knowledge for assembling a complete OSSMM system (headband and Android App). It is out of the scope of this tutorial to explain all the background. Resources will be provided for some areas, but all of the knowledge required to assemble OSSMM can be learned with some patience and simple internet searches.

Building your own OSSMM System requires 4 principal areas of knowledge:

1. **Basic Electronics Knowledge (including how to solder)**
2. **3D Printer Fundamentals**
3. **Android System Familiarity**
4. **Arduino or Microcontroller Programming Basics**

If this seems like a lot, know these are the *general* areas of competency. Specific knowledge within each of these areas with resources is listed below:

What you specifically need to know

1. Basic Electronics Knowledge (including how to solder)

1. What a microcontroller (MCU) and Printed Circuit Board (PCB) are:
    - MCU + PCB Resource (https://www.youtube.com/watch?v=yi29dbPnu28) (Video)
    - Arduino MCUs in 100 seconds (https://www.youtube.com/watch?v=1ENiVwk8idM) (Video)
2. How to Solder
    - Soldering Instructional #1 (https://www.youtube.com/watch?v=6rmErwU5E-k&t=256s) (Video, Note: Excellent instructional. However, We prefer an easier method for pin-hole soldering. This is shown in the Electronics Assembly section.)
    - Soldering Instructional #2 (https://www.youtube.com/watch?v=rK38rpUy568&t=167s) (Video)
    - Arduino Guide to Soldering (https://docs.arduino.cc/learn/electronics/soldering-basics/) (Non-video)

2. 3D Printer Fundamentals

1. How 3D Printers work in principle
    - 3D Printer General (https://www.youtube.com/watch?v=f94CnlQ0eq4) (Video)
2. How to print a file if provided the .gcode files
    - In short, how to put this file on the 3D printer and print it
3. Difference in filament types: PLA and TPU
    - PLA vs TPU Description (https://youtu.be/dYPW5Rlwn8g?t=43) (Video)
4. If your 3D printer is Direct-Drive or not
5. How to adjust your printer to print TPU

3. Android System Familiarity

1. What is Android and familiarity with the UI
2. How to install an APK file (i.e., how to install an app)
    - Installing an APK on Android (https://www.youtube.com/watch?v=Ehlzt2OXl4c) (Video)

4. Arduino or Microcontroller Programming Basics

1. As per the above, what is Arduino and what is an MCU
2. How to upload a sketch from Arduino IDE to an Arduino (i.e., how to program an MCU with existing Arduino code)
    - Learn Arduino in 15 minutes (https://www.youtube.com/watch?v=nL34zDTPkcs&t=93s) (Video)

Build Your Own - What You Need to Have

Required Tools:

Tools:

1. **3D Printer** (Direct Drive Preferred)
2. **Soldering Iron Kit**
3. **Android Device** (which supports BLE, e.g. smartphone)
4. **Computer** (Mac, Linux, or Windows)
5. **12mm Snap Fastener Kit with Setter and Hammer**[1] (with metal snap-fasteners)
6. **Multimeter**
7. **Wirecutters**

Required Expendables:

Expendables:

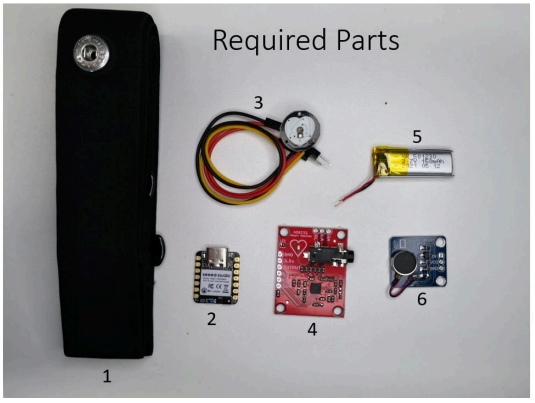
1. **Wires (22-30 AWG)**[2]
2. **PLA 3D Printer Filament**[3] (PLA = Polylactic Acid)
3. **TPU 3D Printer Filament**[3] (TPU-95 or TPU-85, TPU = Thermoplastic Polyurethane)
4. **Solder (lead free, with flux recommended)**

Recommended Tools and Expendables:

Recommended Tools and Expendables:

1. **Helping Hands** (with Magnifier)
2. **Fume Extractor** (if not working in a ventilated environment)
3. **Flux**
4. **Sandpaper**[4]

OSSMM Headband Required Parts:



Parts:

1. **Heart Rate Monitor Strap**
2. **Seeed Xiao nRF52840 Sense**
3. **Pulse Sensor**
4. **AD8232 EKG Board**
5. **3.7V LiPo Battery (120 – 220mAh, not exceeding: L=32, W=15, H=6 mm)**
6. **Vibration Motor Board**
7. **x2 Pair of 12mm Metal Snap-Fastener (not shown, these generally come with the snap-fastener kit mentioned above)**
8. **Sheet of paper (not shown)**

Note: The Vibration Motor Board is only required if you plan to use sleep modulation features. For sleep monitoring only, this component can be omitted.

The Cost

One of the priorities in designing OSSMM was to address the prohibitive costs of sleep research by providing an affordable solution. By leveraging the capabilities of an external Android device[5] it was possible to significantly lower the device cost of the OSSMM headband.

Most tools needed to build the OSSMM are standard equipment found in university laboratories. The only specialized tool required is a snap-fastener installation kit, which costs less than $15 or €12 (as of 12/2024). One kit will provide enough snap-fasteners for more than 30 OSSMM headbands.

3D printing is essential for creating the OSSMM headband. While 3D printers are increasingly accessible in universities and public libraries, it's important to note that a direct-drive extruder is strongly recommended. This type of printer pushes filament directly into the hot end (where filament melts for printing) and is the best method for printing flexible filaments (like the TPU used in OSSMM). While not impossible, printing TPU with other 3D printer types will likely be challenging.

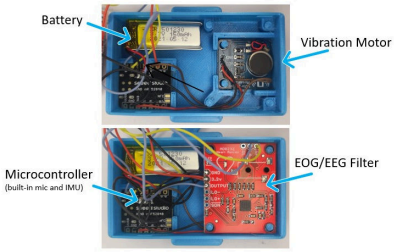
The original target cost for the OSSMM headband was $150. **We achieved a cost of just €37.20 per unit (after rounding up)**. This value represents the marginal cost ("ingredient cost") of building one unit, not the total overhead cost ("shopping list cost")[6].

We estimate the typical marginal cost of building a single unit at €40-60[7][8].

When utilizing "premium" components from established brands without comparison shopping, the total expenditure should still not exceed the original $150 target cost within the USA (shipping included).

This estimate will vary depending on country, suppliers, and brands (and it seems political changes).

Parts Cost



Part	Cost
Microcontroller	€19.00
Battery	€8.00
Silicone Electrode Strap	€2.50
EOG/EEG Filter + Amplifier	€2.50
Pulse Sensor	€2.00
3D Printed Parts	€1.70
Vibration Motor Board	€1.50
Total:	€37.20

When using the OSSMM in formal research settings, regulatory compliance may require technical and safety documentation for all components, including the headband (i.e., heart rate monitor strap). This documentation requirement can affect costs. Currently, only one manufacturer—PolarCare—has provided comprehensive safety information for their Pro Strap, which costs approximately $37 or €36 (as of 04/2025). For comparison, generic commercial heart monitor straps without documentation can be purchased for less than $6 or €5. They can even be purchased for about $1 or €1 in some circumstances. All pricing reflects actual marketplace offerings rather than currency conversions.

The repository will be updated with additional brand headband data and safety documentation as more manufacturers provide this information.

I'm ready - let's begin!

The first physical step to building your OSSMM device is creating the 3D prints. Head over to the 3D Printables (https://jvgiordano.github.io/OSSMM/printables) page to begin!

1. We recommend purchasing the 12mm snap-fastener kit as a whole. The installation tools (setter and hammer) and 12 mm snap-fasteners can be purchased separately but buying the kit together is easiest. ↩

2. We recommend 30 AWG wire that is color coded and silicone insulated. Silicone coatings do not melt at normal soldering temperatures and make assembly cleaner, easier, and safer. Two 5" (13cm) thicker strands, no thicker than 22AWG are recommended but not needed. The whole project can be completed with 30AWG. ↩

3. Filaments should be chosen based on their safety and data profiles. Only the TPU filament will make prolonged dermal contact. Therefore, TPU materials with bio-compatibility documentation should be used. ↩ ↩[2]

4. Sandpaper is recommended for individuals using entry-level or older 3D printers who desire a more refined finish on their prints. Sandpaper may also be beneficial for those who prefer a more forgiving approach to possible electronic component fitting, as sandpaper allows for gradual material removal rather than the harsher scraps or snips using wire cutters. ↩

5. The Android device provides storage, time tracking, and processing capabilities for which it is well suited. Adding these functions directly to the headband would significantly increase the price, size, and weight. As previously mentioned, lower cost alternatives like a Raspberry Pi could be used in lieu of Android devices as a separate standalone "base station" providing these capabilities. We selected Android smartphones due to their widespread availability and familiar user interfaces. ↩

6. Marginal costs refer to the expense of producing one additional unit. For example, we calculate the filament cost per OSSMM headband by multiplying the cost per gram by the grams used in a single headband—not by the cost of the entire filament roll. For a familiar analogy: when calculating the cost of a batch of cookies, we count only the portion of butter used (e.g., half a stick), not the full stick that had to be purchased to make the cookies in the first place. ↩

7. Purchasing some items in bulk quantities may result in lower prices. Our price involved "bulk" quantities where some items were purchased in quantities greater than 10. However, for some market places we have found day-to-day pricing to have a greater effect on pricing than bulk purchasing. It is not necessary to buy in bulk for low-cost pricing. ↩

8. This does not account for fixed costs: equipment, full rolls of solder, filament, etc. ↩

Updated: May 16, 2025

3D Printables (https://jvgiordano.github.io/OSSMM/printables/)

4 minute read

** Difficulty: ★★★☆☆ (Intermediate) **

Welcome to the 3D Printable Guide! Here the basics of creating the 3D prints for the OSSMM are covered.

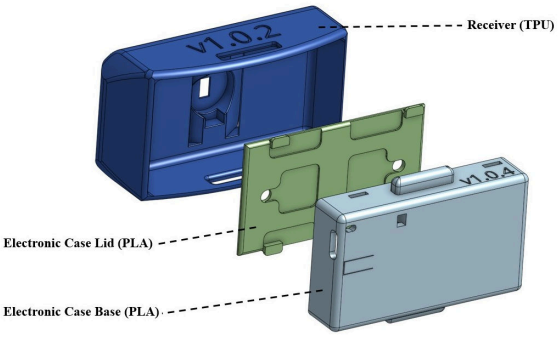
Please note, this is not a general "How To" guide for 3D printing. Because of significant variation in printers, slicing software, and filament choices, it's not possible to create a straightforward, "cookbook" guide. Some information, such as optimal printing temperatures, will need to be determined by yourself.

We recommend following best printing practices. This includes printing a Temperature Tower (https://www.youtube.com/watch?v=NEnVQKX-_N8) to determine the optimal slicing parameters for your specific setup. This step is particularly important if you're working with TPU filament and don't have previous experience with this material in your printer. Other best practices include ensuring filament is kept dried and choosing appropriate supports.

We remind users that direct-drive 3D printers are highly recommended.

Printables Overview

The OSSMM requires three 3D-printed components: the receiver, the electronic case lid, and the electronic case bottom. A labeled CAD image of these is shown below:



Labeled OnShape CAD for the OSSMM 3D Printables

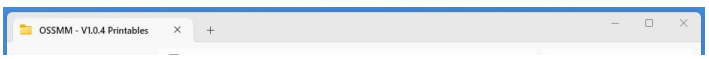
As denoted in the image, and previously mentioned, the receiver is printed using TPU (Thermoplastic Polyurethane), a flexible filament. The electronic case is printed in using PLA (Polylactic Acid).

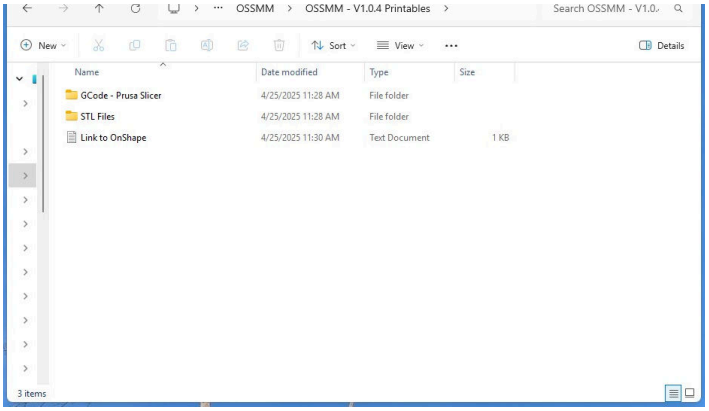
Because these components use different filament types, the receiver must be printed on a separate print run from the electronic case's top and bottom (unless you have a multi-filament 3D printer with independent extruders). Therefore, there is one STL file for the receiver, and one STL file for the electronic case.

All three parts are drawn in a publicly available OnShape Document located here (https://cad.onshape.com/documents/9770a63668febc8c28355357/w/080415be9bd2ce05fb700580/e/ca5bfcf86b2d6d5dd97a01c1?renderMode=0&uiState=680bf8be89fa246ebf943572). You can fork this CAD document and modify as needed.

3D Print File - STLs and GCODE

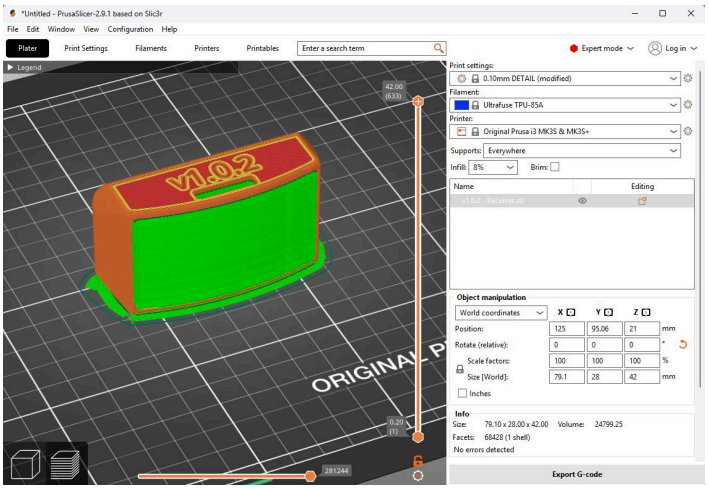
The STL file for the receiver and electronic case is located in the OSSMM Github repository under "STL Files" within the OSSMM - V1.0.4 Printables (https://github.com/jvgiordano/OSSMM/tree/main/OSSMM%20V1.0.4%20Printables) folder.





3D Printing Files in the OSSMM Github Repo

The STL files will need to be sliced by your slicing software of choice, and then the Gcode loaded to your 3D printer for printing.



Receiver in PrusaSlicer with "snug" supports and 8% infill.

For (possible) convenience, we have provided the GCode from Prusa Slicer that we used on a Prusa MKS3+ 3D printer. However, it is still recommended to go through the proper 3D printing best practices and starting with the STL files and adjusting the slicer parameters to your own printer (even if you own a MKS3+ and use the same filament as us).

Printing the Receiver

Filament

Print the receiver with TPU grades from 85 to 95 shore A hardness (TPU-85 to TPU-95). Softer filaments may work but have not been tested.

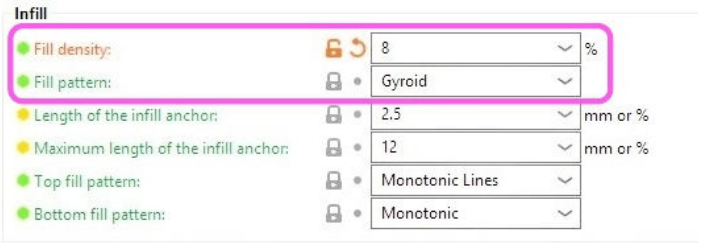
We recommend TPU-85 for optimal comfort, though TPU-95 works sufficiently well. Be aware that TPU-85 is more challenging to print due to its softness compared to TPU-95.

We used Siraya Tech Flex TPU85A filament because of safety considerations. Since the receiver makes prolonged dermal contact, we prioritized Siraya Tech for their multiple biocompatibility certifications. These certifications can be found on the company's website or in the OSSMM - Data and Safety docs (https://github.com/jvgiordano/OSSMM/tree/main/OSSMM%20-%20Data%20and%20Safety%20docs) folder in the OSSMM repository.

Slicer Settings

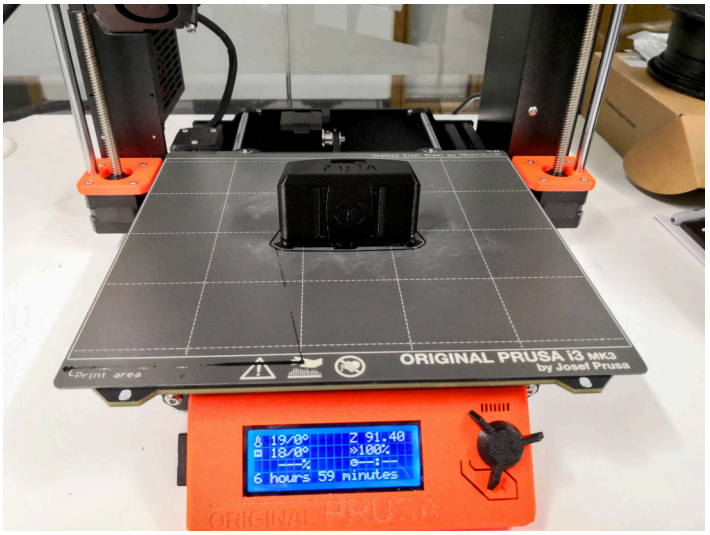
Important: When printing with TPU, two critical slicer settings must be configured:

1. **Set “Infill Density” to 8%**
2. **Set “Infill Pattern” to “Gyroid”**



Required Infill Settings for TPU-85

The receiver was specifically designed using TPU with these infill settings to create an “air cushion” effect that comfortably distributes pressure. If printing with TPU-95 (higher hardness), we recommend reducing the infill percentage to 7%. Optimal values may vary depending on your specific printer, filament brand, and slicer parameters such as flow rate. The goal is to create a receiver that provides soft cushioning for comfort without collapsing or breaking.



Printed Receiver on MK3S+ using Siraya Tech TPU-85A with a print time of nearly 7 hours.

Printing the Electronic Case

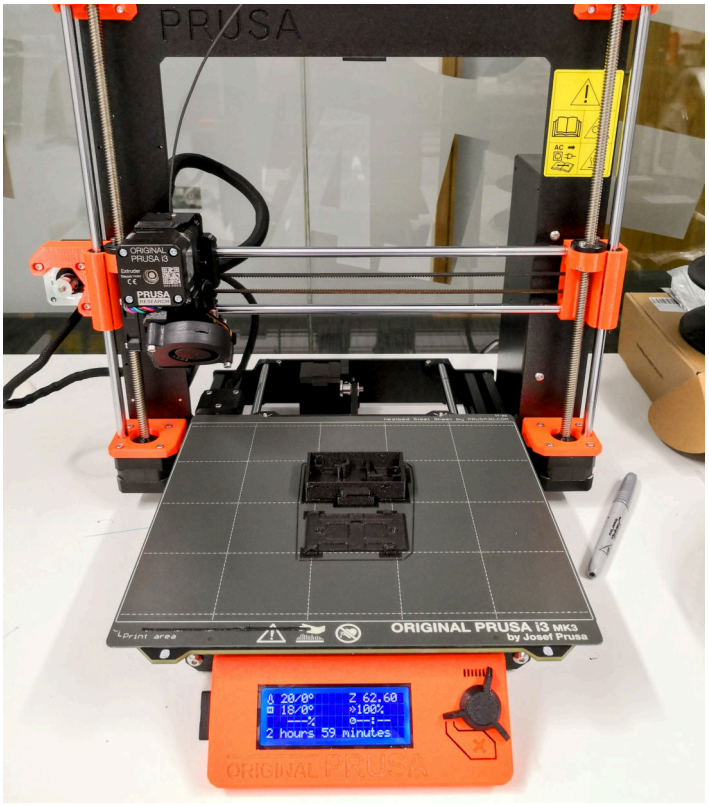
Filament

Print the receiver with standard PLA.

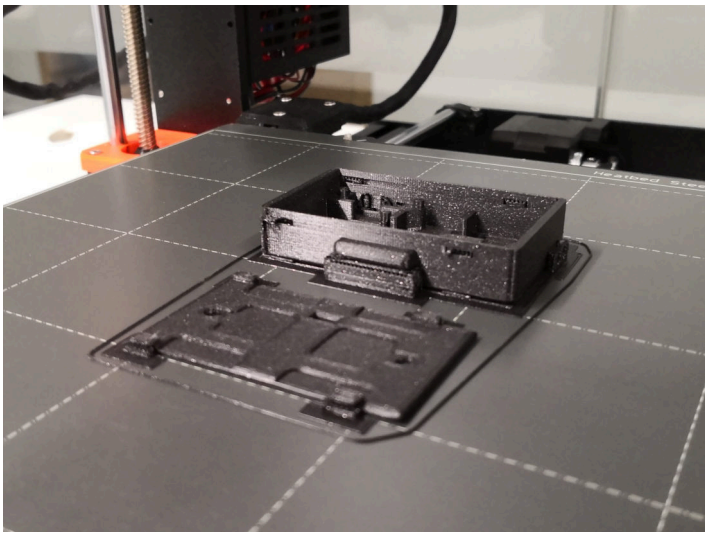
We used Prusament PLA for both its fine printing quality, and its safety considerations. Although PLA material does not make any prolong dermal contact, safety is still a priority and Prusament provides suitable safety information.

Slicer Settings

The electronics case is a typical 3D print and there are no required slicing parameters. Because of the fine structures within the case we recommend printing at the highest printing quality possible.



Printed Electronic Case on a MKS3+ in Prusament Galaxy Black with a print time of nearly 3 hours



Close Up of Print Electronic Case

Finishing 3D Prints

The 3D prints should be inspected, and any support material, stringing, or printing burrs removed. Light sanding with sandpaper may be used to improve the finish of prints. In particular, the rear of the receiver should be smooth.

If glue was used for bed adhesion, make sure to properly clean any residue off the prints. We recommend avoiding glue for bed adhesion where possible.

Time for the Electronics Assembly!

The next step to building your OSSMM device is assembling the electronics. Head over to the Electronics Assembly (https://jvgiordano.github.io/OSSMM/electronics-assembly) page to begin!

Updated: May 16, 2025

Electronics Assembly Guide (https://jvgiordano.github.io/OSSMM/electronics-assembly/)

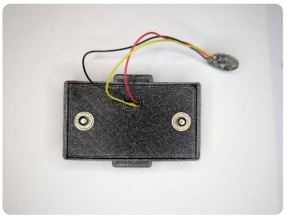
16 minute read

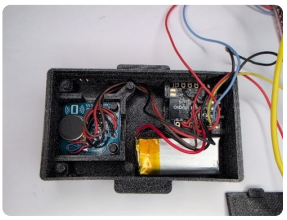
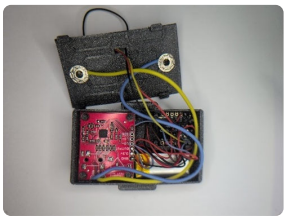
** Difficulty: ★★★★☆ (Challenging) **

Welcome to the Electronics Assembly guide! In this section, you will learn how to combine the electronic components and install them in the 3D printed case to create the “electronic module.”

The photos below show the completed assembly from different angles. Some images display the case with the lid open to reveal the internal component arrangement. Please note that the PulseSensor intentionally remains outside the electronic case in the final assembly.

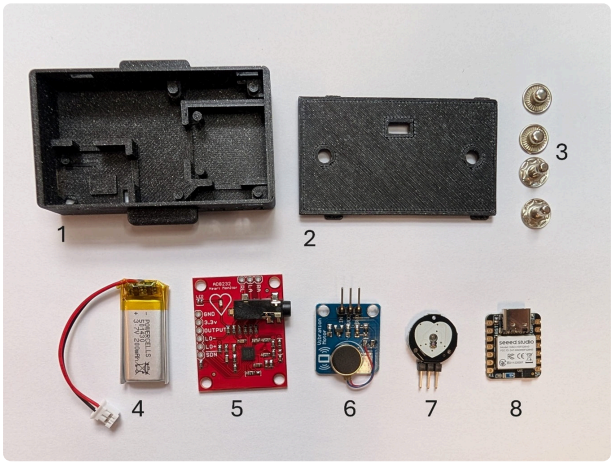
Build Time:
- First-time: expect 2-2.5 hours
- Experienced: comfortably under 50 minutes

Note: Edges may appear curved in some photos due to lens distortion during the documentation process. Please be aware that these edges, like the sides of the electronics case, are actually straight.

Step 1. Gather Relevant Tools and Materials:



Parts:

1. **3D Printed Electronics case Bottom**
2. **3D Printed Electronics case Lid**
3. **x2 Pair of 12mm Metal Snap-Fastener (2 male, 2 female)**
4. **3.7V LiPo Battery (120 – 220mAh, not exceeding: L=32, W=15, H=6 mm)**
5. **AD8232 EKG Board**
6. **Vibration Motor Board**
7. **Pulse Sensor**
8. **Seeed Xiao nRF52840 Sense (MCU)**
9. **Sheet of paper (not shown)**

Required Tools

1

2

3

4

Required Tools:

1. **Soldering Iron**
2. **Snap Fastener Hammer and Setter**
3. **Wire Cutters**
4. **Multimeter**

Recommended Tools

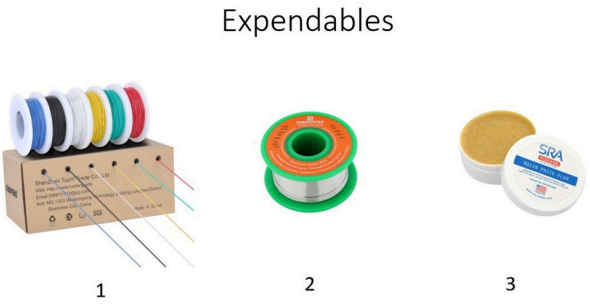
1

2

Recommended Tools:

5. Helping Hands with Magnifier

6. Fume Extractor (if not working in a ventilated environment)



Expendables:

1. **Wires (30 AWG is recommended with color coded, silicone coating. We also recommend two 5" (13cm) thicker strands, no bigger than 22AWG. The whole project can be completed with 30AWG however)**
2. **Solder (lead free recommended)**
3. **Flux (particularly if not in solder)**
4. **Sandpaper (Recommended)**

Preliminary notes on soldering:

1. **Soldering temperature should be set to 380-400°C.**
2. **When using solder, especially leaded solder, use appropriate ventilation and wash hands afterwards.**

Please refer to this Arduino documentation on soldering (https://docs.arduino.cc/learn/electronics/soldering-basics/) for best practices and more information.

Step 2. Test-Fit Electronic Components in 3D Printed Case Bottom

Goal: Verify that all electronic components properly fit into the 3D printed case bottom.

In this step, you'll test if your components fit correctly in the case.

Refer to the photos directly below to identify the position of each component. Don't force anything into place at this stage—we're only checking fit, not installing components permanently.

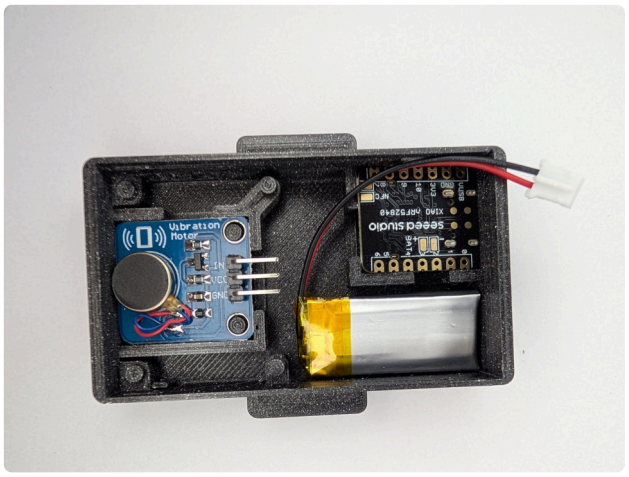
If components appear too large:

1. Check if the case was printed to correct specifications
2. If the case dimensions are correct, then some PCBs* may need minor adjustments
3. Carefully sand edges of PCBs using sandpaper or scrape with wire cutters (use caution!)

Important fit check: The AD8232 and Vibration Motor boards have holes that should align with the pegs in the case. This alignment should confirm proper case dimensions.

*PCBs (Printed Circuit Boards) refer to the component boards such as the Seeed Xiao Sense, Vibration Motor Board, AD8232, and Pulse Sensor.

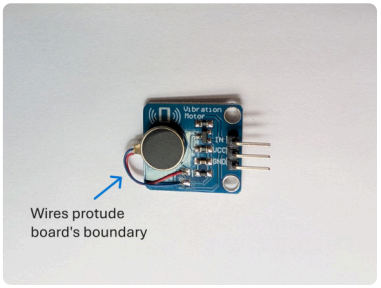
The photo directly below shows a properly positioned Vibration Motor, along with the designated positions for the battery and MCU.

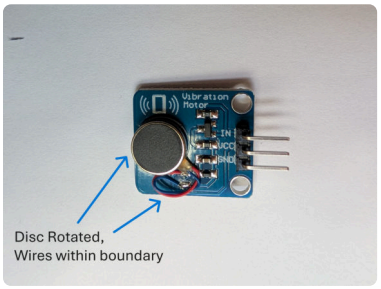


Please note that the Seeed Xiao Sense inserts bottom-side facing up, so that the USB port inserts into the electronics case's USB slot.

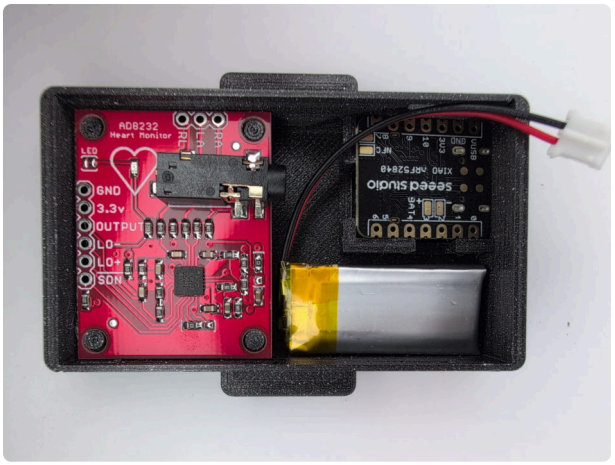
The Vibration Motor Board may need minor adjustment to fit properly. The vibration disk's orientation might cause wires to extend beyond the board's boundaries, preventing insertion into the electronics case. Simply lift the vibration disk from the board (it's attached with double-sided adhesive) using the wire cutters or your fingers, reorient it so wires stay within the board dimensions, and press it back down gently to reattach.

The following photos highlight this issue. The left image shows the problem: the vibration disc orientation causes wiring to extend beyond the board's boundary. The right image shows the solution: rotating the disc keeps all wiring contained within the board dimensions.





The photo below shows the inclusion of the AD8232, which should sit above the Vibration Motor Board.



If the components fit, move on to the next step. Otherwise, take the time to reprint the case or gently modify the PCBs as needed.

Step 3. Connect Snap-Fasteners to the Electronics case Lid

Goal: Install Snap-Fasteners on the Electronics Case Lid with Wire Leads

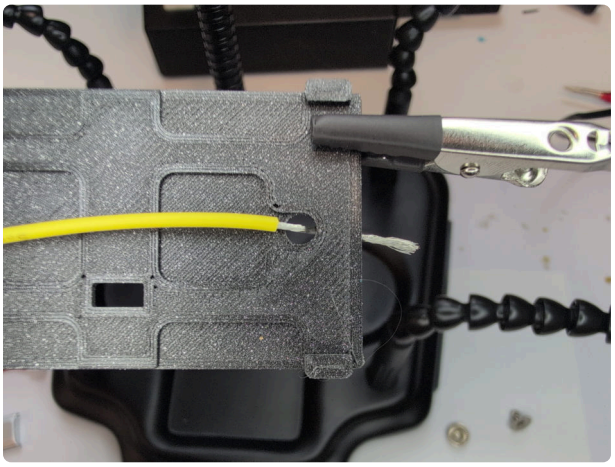
The snap-fasteners function as electrode connection points for the heart rate monitor strap, which collects EOG and EEG signals. This connection is essential for accurate sleep monitoring. Take your time with this step.

Materials needed:

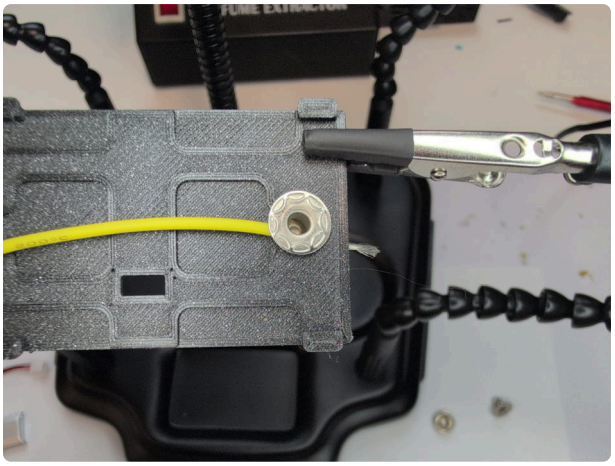
- Two wire strands (22-30 AWG), at least 5" (13cm) long
- Thicker wire (22 AWG recommended) provides better electrical contact

Preparation steps:

1. Strip approximately 20-25mm of insulation from one end of each wire
2. Locate the internal face of the electronic lid (the non-smooth surface)
3. With the internal face up, thread the exposed wire through one of the round holes
4. Position the wire carefully in the notch as shown in the photo below:

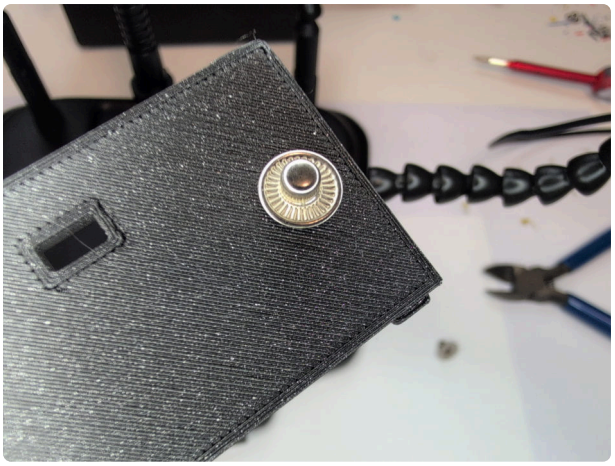


Then, insert one of the male snap-fasteners into the same hole.

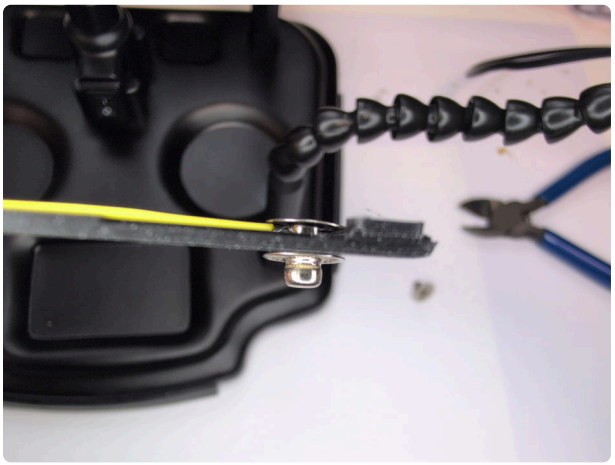


Flip the electronic lid over. Bend the exposed wiring over the male snap fastener. Trim excess wire.

Next, place a female snap-fastener over the male snap fastener **while maintaining the exposed wire's position**. It's normal for some wire strands to shift slightly during this process—as long as most of the wiring stays in place, the connection will function properly.



Press the snap-fastener assembly together firmly by hand. This creates an initial connection that prevents the wire from easily pulling free. You'll notice small gaps between the snap-fastener assembly (as shown in the photo below). Don't worry — we'll address this in the next step.



To close these gaps and create a secure electrical connection, we'll now properly crimp the snap-fastener. Gather your snap-fastener setter and hammer. The setter is the metal rod with a concave end which is shaped to fit over the female end.

Find a sturdy corner of a hard, flat surface for this step. Using a corner is important as it provides targeted support directly under the snap-fastener while keeping the electronics case lid's side tabs free from pressure that could damage them.

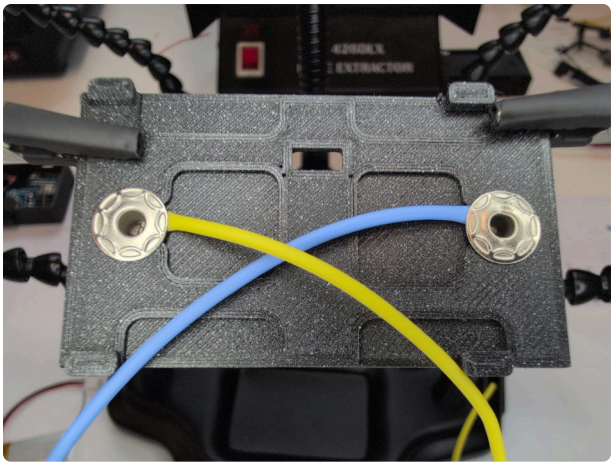
First, verify the wire remains in the notch on the internal face. Then position the electronics case over the corner so that only the male end of the snap fastener receives support from below, without the corner touching any other part of the case lid. This positioning will prevent any damage to the four side tabs on the lid.

Place the setter tool directly on top of the female snap fastener and strike firmly with the non-rubber side of the hammer. Don't hesitate to use sufficient force or multiple strikes to achieve a proper crimp—this connection needs to be secure.

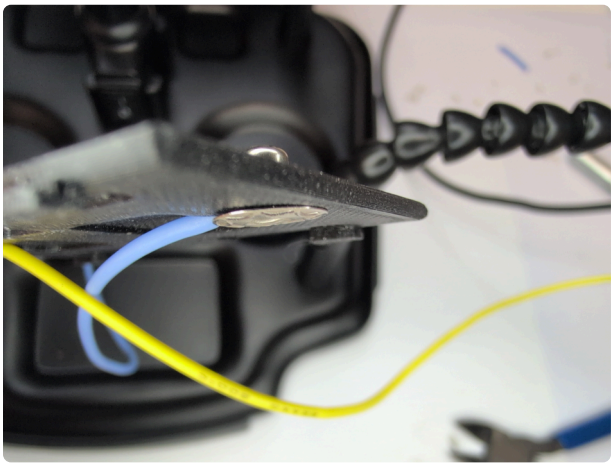
While it is possible to over crimp the snap fastener and cut the wire, this is difficult to achieve if the wire is positioned in the notch from the start.

Remember, the goal is to have a tight, secure fit that will not come undone and will make solid contact with the exposed wiring. This is the conductive path for the EOG and EEG signals and it is essential for accurate sleep monitoring.

Repeat the process for the other hole. Make sure the snap fasteners are secure. Below is a complete electronics case lid with attached snap-fasteners:



It is okay to have a little gap as shown below if the connection is snug. Again note that over crimping can lead to the wire being cut, but if the wires are placed correctly in the notch this is unlikely.



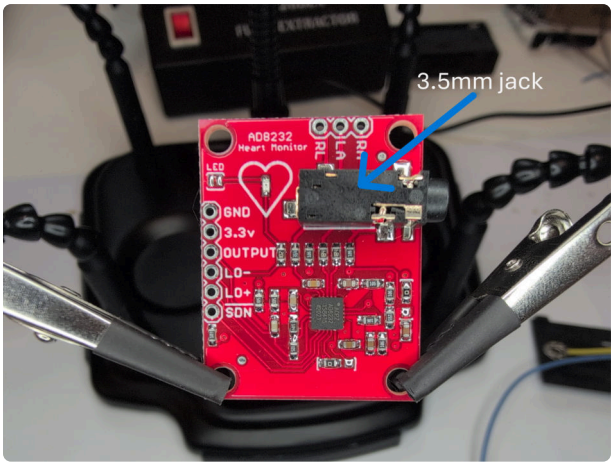
Step 4. Prepare the AD8232

Goal: Begin preparing the PCBs for connection with the MCU. First PCB to prepare is the AD8232.

Before assembling all components together, each circuit board may require some preparation. We'll start with the AD8232 EKG board.

The AD8232 EKG board includes a 3.5mm headphone jack that we won't be using in our design. Since we'll connect directly to the board's RA and LA pins (leaving the RL pin unused), this jack must be removed to allow the board to fit properly inside the electronics case.

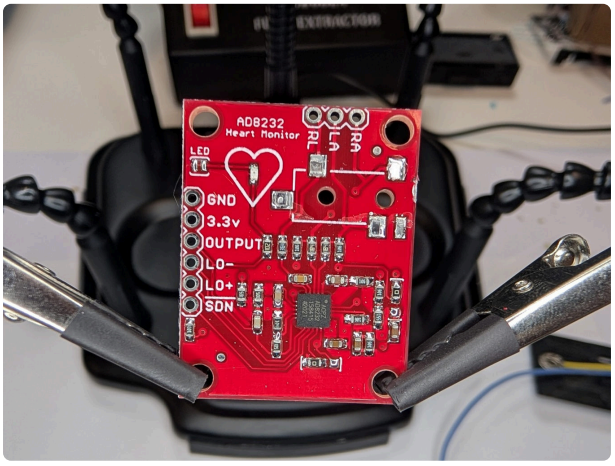
For those interested in more information about this design choice, please refer to the section below titled, Additional Information on the AD8232 below



Quick Method: Use wire cutters to firmly pry the jack off the board. Don't hesitate to apply reasonable force.

Alternative Method: If you plan to repurpose this board later, consider desoldering the jack instead. Prying may occasionally detach electrode contacts, though this won't affect our current application.

Note on Reliability: We've tested this removal method on numerous boards and found prying to be both quick and reliable, with no board failures to date.



<img src="https://jvgiordano.github.io/OSSMM/media/electronics-assembly/JackRemoval.gif" style="width: 50%; alt="3.5mm Jack Removal Process" style="display: block; margin-left: auto; margin-right: auto;">

Once the 3.5mm jack is removed, the AD8232 board should appear as in the photo below:

Step 5. Prepare the PulseSensor

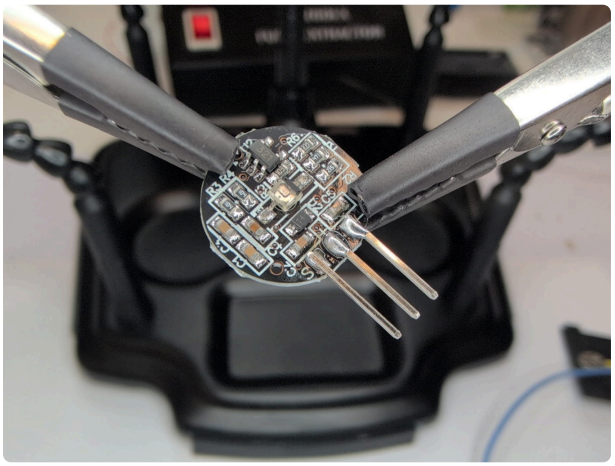
Goal: Prepare the PulseSensor board.

PulseSensor units will either come soldered with pins or with wires. The OSSMM requires wires.

Even if the unit comes with wires it is recommended replacing them with high-quality silicone-insulated wires. This will prolong the durability and longevity of the device, and make later assembly easier.

Note: Future designs may permit the PulseSensor to only require pins. Wires coming from the MCU could terminate with female Dupont connectors which would mate with these pins. This would have several benefits. However, the current design does not lend itself to this.

If you have pins, it is easiest to remove the black separator then desolder the pins one by one.

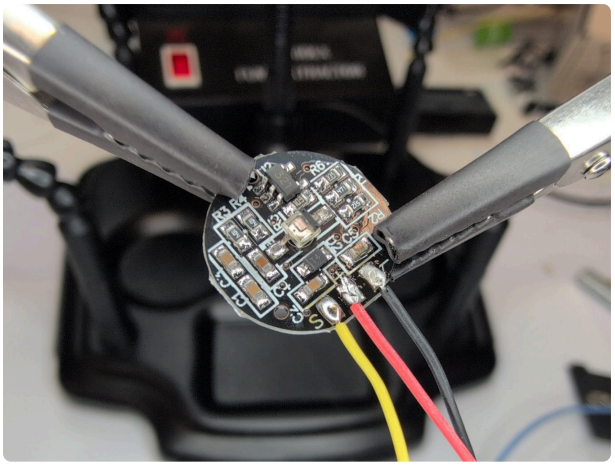


Wiring Decision: You can add wires to the PulseSensor now or later.

Both approaches work, but **we recommend adding the wires later.** The PulseSensor's wires must pass through the rectangular slot in the electronic case lid. Adding the wires earlier makes the soldering more challenging.

Regardless of when you choose to add the wires, the photo below shows the correct wiring configuration. When rewiring please confirm that the order of the electrodes of:

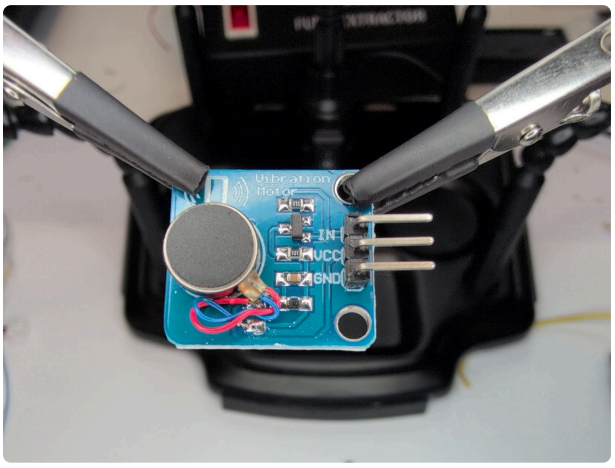
- **S** (Signal)
- **+** (3.3V)
- **–** (Ground, 0V):



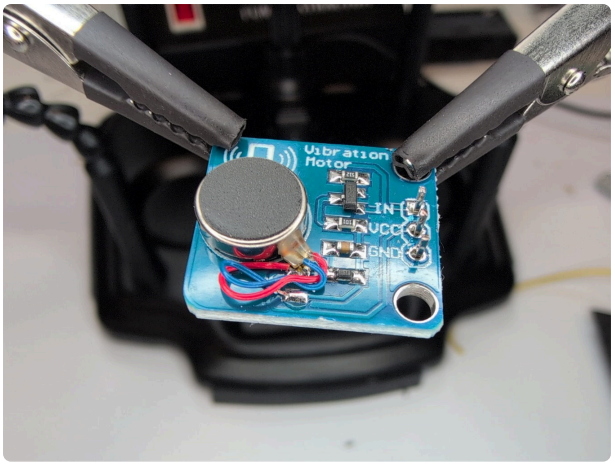
Step 6. Prepare the Vibration Motor

Goal: Prepare the Vibration Motor Board

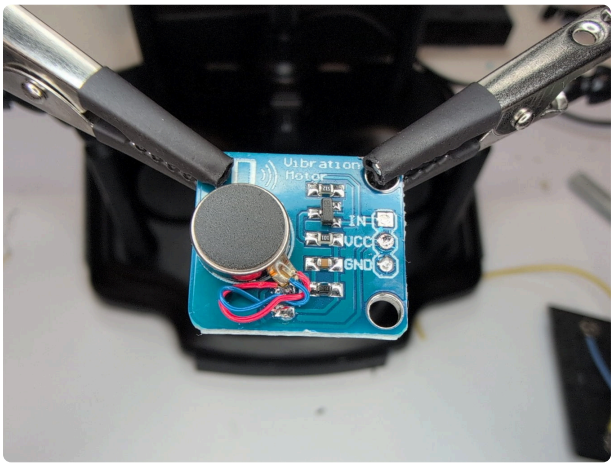
If the vibration motor board comes with pins these must be removed.



If the pins are bent, we recommend cutting below the bend, then remove the plastic separator as shown in the photo below**. It is not recommended to cut the pins as low as possible on the board.** They are easiest to remove with tweezers or pliers while desoldering if there is something to grasp.



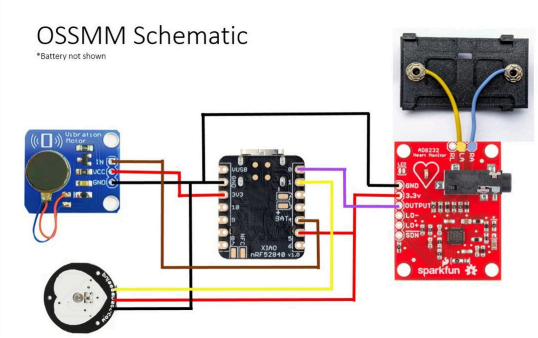
The prepared board should look as the below. Nothing external protruding over the board's boundaries, and electrode pinholes ready for wires.



Step 7. Solder Components Together According to the Layman's Schematic

Goal: Assemble all the components!

It is now time to assemble all the components with wires! Refer to the Layman's schematic below for the connections. We will show step by step soldering photos further below:



Before proceeding, please note the following:

1. **MCU:** All wires exit from the bottom face of the board.
2. **AD8232 Board**:
    - Side Pins (GND, 3.3V, Output): Wires exit from the bottom side (opposite face of the 3.5mm jack).
    - Top Pins (RA, LA): Wires exit from the top side (face with 3.5mm jack).
3. **Vibration Motor Board**: Wires can exit from either side. Top side is preferred.
4. **PulseSensor**: If you have soldered wires to the PulseSensor, remember to thread the wires through the rectangular slot in the electronics case lid before soldering to the MCU!

We will remind you of these again later in the tutorial.

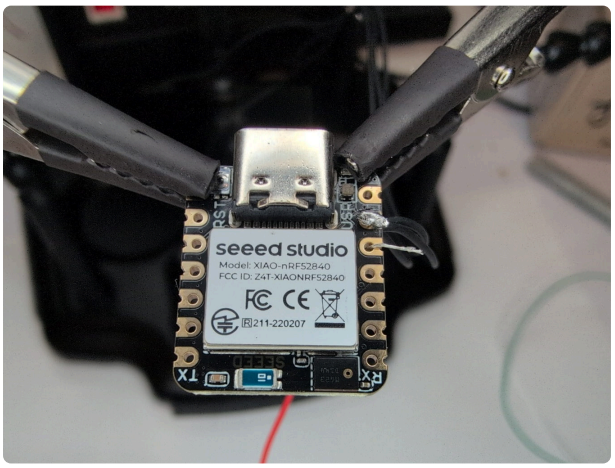
There are many ways to go about connecting the components with the schematic above. We relate here the method we've found most reliable and efficient:

Begin by soldering all required wires to the appropriate pins on the MCU board:

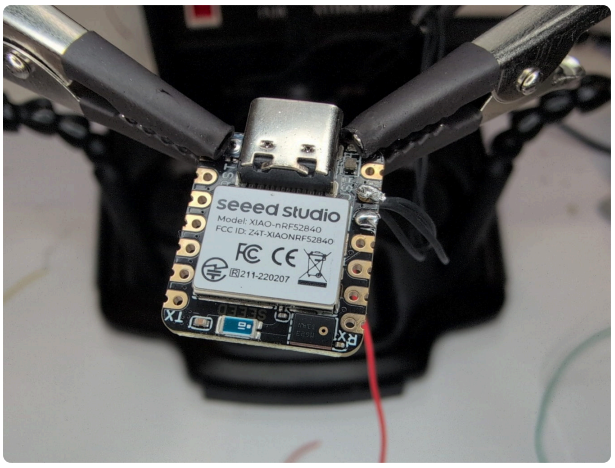
1. Thread each wire through its designated pin hole from the bottom side of the board
2. Bend the wire on the top side to hold it in place
3. Solder each connection securely
4. Trim and clean off any excess

Shown below is the first and second step of this process. Here we have threaded a red wire through the 3.3V pin of the MCU which will later power the Vibration Motor. Already soldered are the three ground wires for the component PCBs.

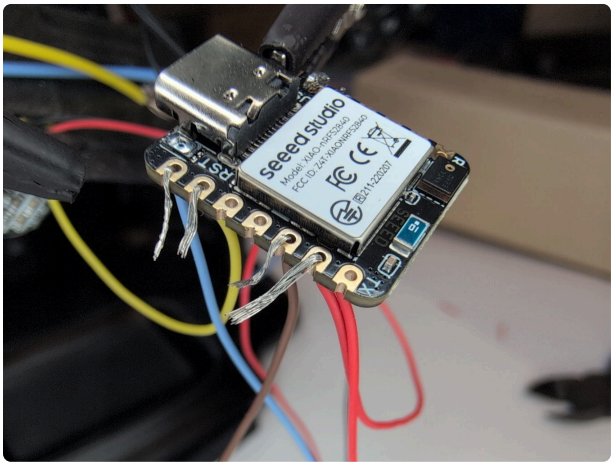
We highlight again, upon completion wires must emerge from bottom of the board:



Now it is time to solder the bent wire, and trim off any excess. Resoldering may be done for a cleaner job.



Now you should connect the wiring for the other side of the board as shown below. Refer to the schematic if there is any confusion. Again we have threaded the wiring through the pin holes and bent the wires so they hold themselves.



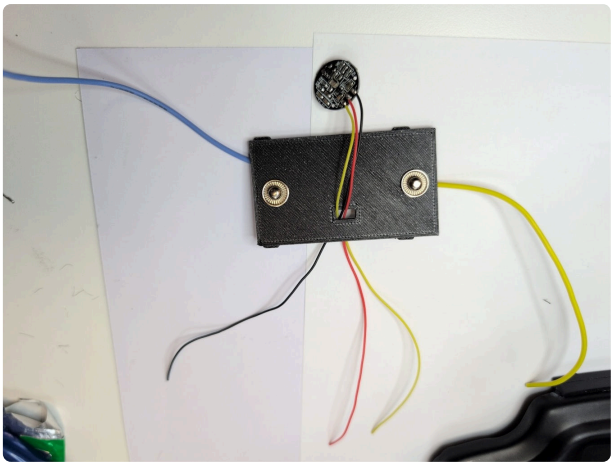
Solder these connections then trim any excess.

Now it is time to complete the connection between the MCU and the PulseSensor.

We highlight again here that the wires for the PulseSensor must go through the rectangular slot in the electronics case lid. If you forget to do this, don't worry. Simply desolder the connections on the PulseSensor, thread the wires correctly, then resolder.

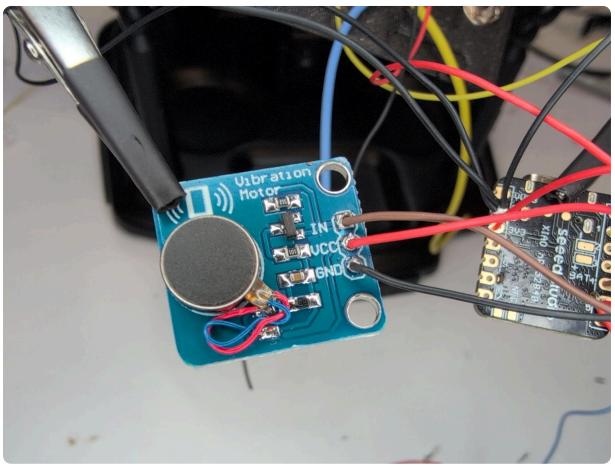
We do not recommend resoldering the connection from the MCU side as the pins are very small and not conducive to easy resoldering.

Below is an example demonstrating the PulseSensor with wires properly threaded through the rectangular slot in the electronics case lid. Again this photo is for illustrative purpose and we recommend connecting the PulseSensor to the wires after they have already been attached to the MCU.

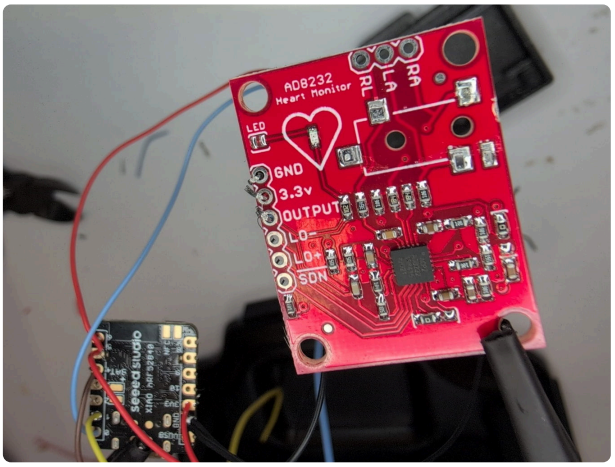


Please fully connect the PulseSensor to the MCU now.

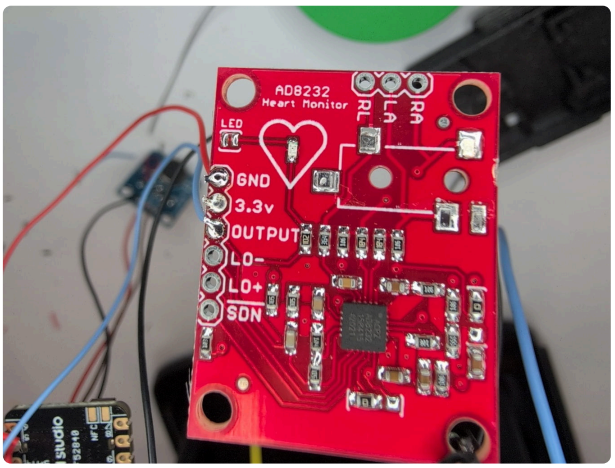
Now you will connect the Vibration Motor Board as shown below:



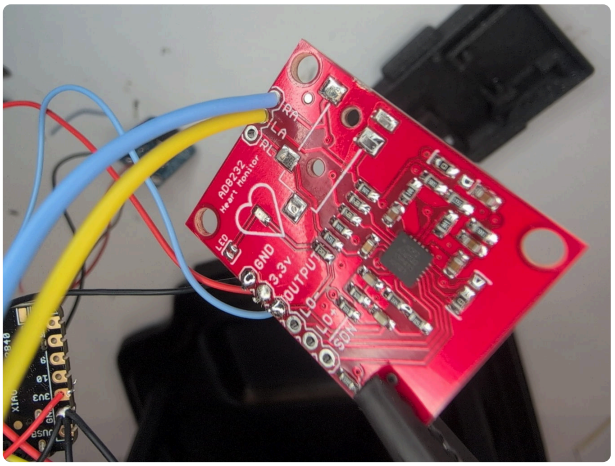
Finally, connect the AD8232 EKG Board. Start with the side connections (GND, 3.3V, and OUTPUT). Remember that these wires must exit from the bottom of the board (the side opposite where the 3.5mm jack was removed). Thread the wires through the pin holes as shown in the image below.



Next solder then clean the connections by snipping off any excess.



Now, solder the RA and LA connections using the two wires from the electronics case lid. In this tutorial, we've used thicker yellow and blue wires for these connections. Either wire can connect to LA or RA as the assignment doesn't matter. Remember that these wires should exit from the top of the board (the side where the 3.5mm jack was removed).

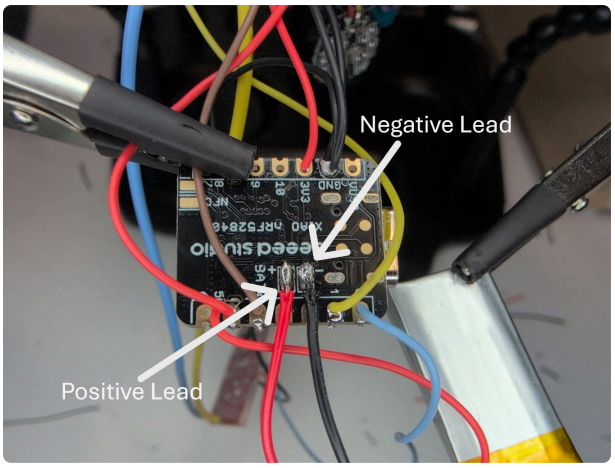


Step 8. Connect the Battery

Goal: Connect the Battery!

Now it's time to connect the battery.

⚠️ **CRITICAL SAFETY CHECK** ⚠️
IMPORTANT: YOU MUST VERIFY BATTERY LEAD POLARITY WITH MULTIMETER
- RED should be POSITIVE (+) - BLACK should be NEGATIVE (-)
CAUTION: Some manufacturers reverse this standard color coding.
⚠️ **CONNECTING INCORRECT POLARITY WILL PERMANENTLY DAMAGE THE MCU** ⚠️
With the polarity of the battery leads verified, **carefully solder the leads to the MCU battery contacts** as shown in the photo below. Here RED is positive, and BLACK is negative.



Accidentally soldering the leads to the wrong location (such as positive to negative) will likely damage your board. The extent of the damage may vary, but if this occurs the MCU should not be trusted even if it appears to function.

Our testing has shown that sometimes the MCU will semi-function, but other errors may present themselves in time, such as the MCU being incapable of charging the battery.

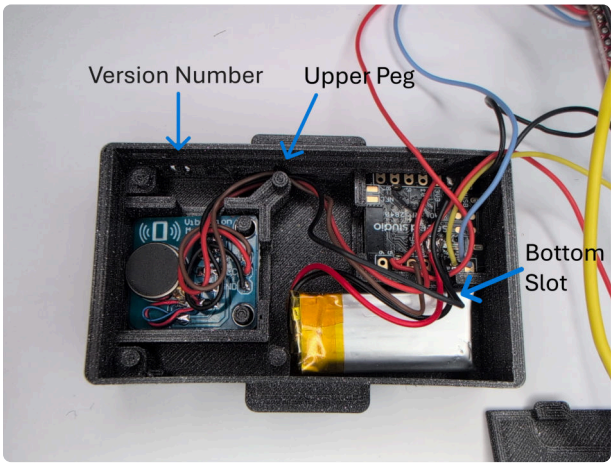
Step 9. Install Components into the Electronics case Bottom

Goal: Slot all electronic components and wiring into the electronic case bottom.

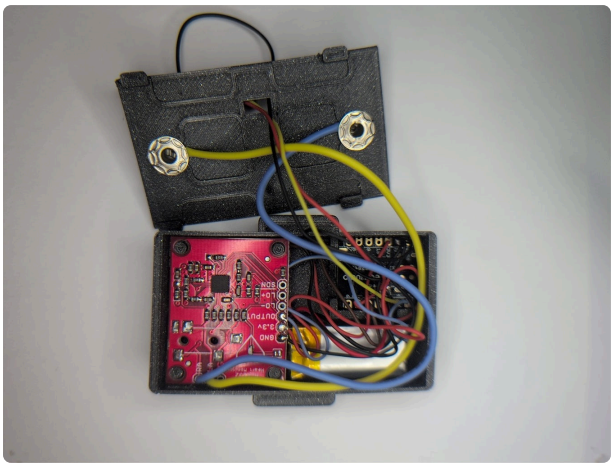
Now that everything is connected, it's time to install the electronics into the case bottom.

First, position the case bottom so you're looking into it. Verify the version number (printed on the side) is in the upper left corner. The MCU slot should be in the upper right corner.

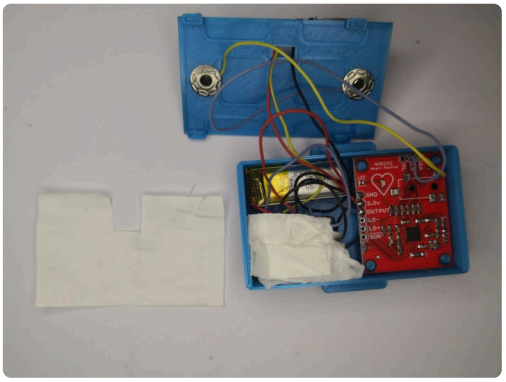
Install the MCU (upside down), the battery, and Vibration Motor Board. Loop the Vibration Motor Board wires around the upper right peg as shown below. Likewise, the battery wires should thread through the bottom slot of the MCU enclosure.



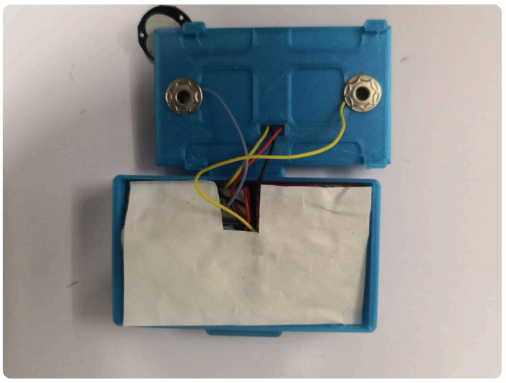
Next, install the AD8232 Board into position. The side pins should be located over the center of the electronics case, and the EOG/EEG (RA/LA) pins should be towards the bottom left corner.



If the MCU is overly loose in its slot, we recommend covering it with folded paper as shown in the figure below to ensure it remains in its slot during use. Please note the following photos show a separate OSSMM device printed in blue, and with thinner 30AWG wiring for the EOG/EEG electrodes.



We also recommend installing one sheet of paper cutout as shown in the photo below over the circuits. This provides an insulative barrier from the metal snap-fastener electrodes and eliminates any possibility of shorting with the underlying PCBs.



Step 10. Close the Electronics case

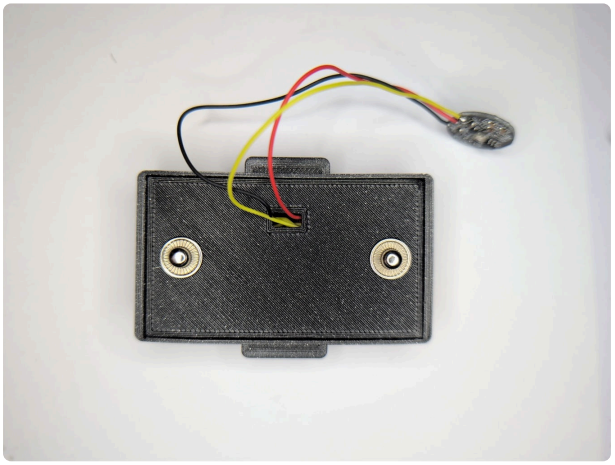
Goal: Close the case and finish!

Now it's time to close the electronics case. Note that the tabs on the electronics case lid are asymmetrical. It can only be closed in one way!

Before inserting, make sure the respective slots on the electronics case bottom are free and clear. Remove any debris or support material which may remain.

Align the tabs, insert on one side, then insert on the other. This may require some slight bending of the electronics case walls to allow the tabs to slip into their slots.

When it is finished it should look like the following:



Back

Front

Side with USB-C

Step 11. Time for Full Assembly!

Refer to the assembly guide below on how to combine the heart monitor strap, receiver and electronics module together! The biggest challenge is over!

Final Assembly (https://jvgiordano.github.io/OSSMM/final-assembly/)

Additional Information on the AD8232

Regarding the removal of the 3.5mm jack, one of the project goals was to minimize the size and weight of the OSSMM device. Under this lens, the 3.5mm jack provides no benefits and consumes precious space resources.

For those familiar with the AD8232, no connection to the RL pin indeed means the Right Leg Drive Amplifier (RLD) is not used. Typically, the RLD applies a small current which is inverse to the signal commonly detected by both the RA and LA electrodes. Because the AD8232 acts as a differential amplifier in this case, applying the inverse of the common signal improves the Common-Mode Rejection Ratio, or in short, improves the Signal to Noise Ratio (SNR) of the signal difference between the electrodes.

We accept the trade-off of reduced signal quality by abandoning the RLD for two principal reasons: perceived safety and constructability. Having any current applied to the body, particularly the head, is considered invasive and therefore makes any risk however infinitesimally negligible, non-zero. Furthermore, the requirement for a third electrode to supply this current to the body increases device costs and device complexity. The performance of the OSSMM without the RLD is more than sufficient to gladly accept this trade-off.*

**To be very clear on how safe the RLD already is, please consider the following: The AD8232 EKG board in its standard "Cardiac Monitor configuration," which is used here prior to "safety" modification, is set with an "RL" output resistor of 360kΩ limiting the output current to a participant's body of ≤ 10 µA, provided the PCB voltage is 3.3V. In the worst case scenario, even if the voltage were raised to 4.2V (i.e. directly powered by a fully charged 3.7V LiPo battery, which it is not) and the resistor had the lowest value within the ±10% tolerance range (that is 324 KΩ), the maximum current would be 13 µA. In comparison, tACS and tDCS devices supply current to a participants head of up to 2mA, or more than 150x the worst case scenario of the AD8232. However, given that there is still a "risk" using the RLD, this would needlessly complicate Health&Safety assessments and ethics approval.*

Updated: May 16, 2025

Major Parts Assembly (https://jvgiordano.github.io/OSSMM/final-assembly/)

8 minute read

** Difficulty: ★★☆☆☆ (Beginner) **

Now it's time for the final assembly! You should have before you the three major components for the OSSMM headband:

1. The strap
2. The electronics case
3. The receiver

Before we can assemble these components, we need to modify the heart-rate monitor strap. Since it's designed to fit around the chest, it's too long for use as a headband and must be shortened. Then we will connect the components together easily!

The modular design of the OSSMM system offers important benefits for future use. If any issues arise with the OSSMM headband, these principal components can be quickly replaced rather than the entire system being changed. For example, the strap can be removed for washing or replaced entirely if needed.

Now let's combine the major components and create an OSSMM headband!

Prepare the Strap

In this section we will cover checking your strap for suitability with the OSSMM system, then guide you through shortening it. Because heart-rate monitor straps vary in build, please consider this a general guide. Your strap's buckle implementation may differ slightly from our examples.

Step 0. Important Note on Strap Selection!

Not all heart-rate monitor straps on the market are equivalent. Importantly, the conductance of the silicone electrodes varies significantly between brands, which affects the Signal-to-Noise Ratio (SNR) of the EOG/EEG signals.

To clarify what this means in practical terms: SNR describes how easily the electrodes transmit the signals we want compared to unwanted electrical noise. Lower electrode resistance (i.e., higher conductance) results in better SNR. Better SNR means cleaner signals and more accurate data capture and sleep staging by the OSSMM system.

Unfortunately, it is difficult to know these conductance (or resistance values) without the manufacturer's data sheet. Polar Care has provided these details for their Polar Pro Strap, specifying a surface resistance of 500 Ω.

You can measure an approximate surface resistance of your strap's electrodes using a quality multimeter:

1. Set the multimeter to resistance measurement mode (2kΩ setting if applicable)
2. Place probes at a fixed distance on one silicone electrode
3. Apply sufficient pressure for good contact
4. Record the reading
5. Repeat for the other electrode to confirm repeatable measurement

This measurement will give you an estimate of your strap's conductivity. Please be aware that true surface resistance measurements require specialized equipment. It is recommended where possible to find heart-rate monitor straps with the lowest resistance.

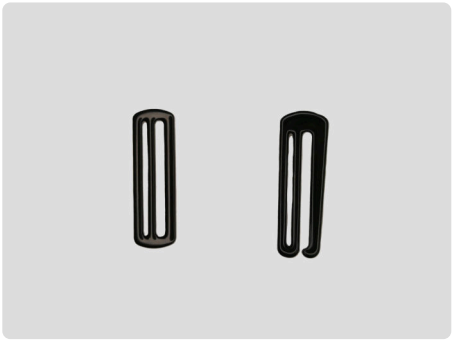
Technical validation for sleep staging found that even electrodes with an approximate surface resistance of 620 Ω were adequate for EEG signature detection with machine learning classifiers. Some brands produce straps with electrodes with resistance levels as low as 135 Ω.

While most commercially available straps should provide sufficient conductivity, we recommend testing

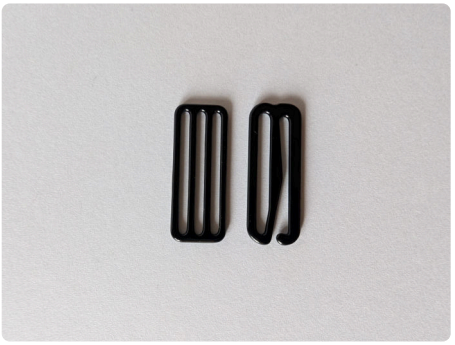
your strap in advance if you don't have access to its data sheet.

Step 1. Identify the Buckle Types on Your Strap

This guide focuses on straps that use either a triglide slide (3 bars) or quad slide (4 bars) in combination with a G Hook slide, as shown in the images below. Examine your strap to identify which type of slides it has. You don't need to remove the slides for this step.



Slides in strap: Triglide slide on left, G hook slide on right



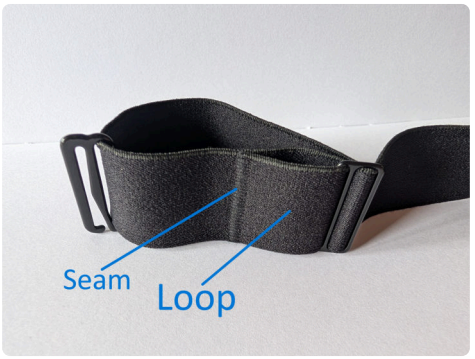
Slides in strap: Quad slide on left, G hook slide on right

The following example uses a quad slide and G Hook slide. If your strap has slightly different slides, use this guide as a general reference for making adjustments to your specific strap.

Step 2. Locate the Slide Types and Stitched Loop

Cinch your strap so that both the G Hook Slide and Quad/Triglide slide are closer together, as shown below. Position them in the same orientation as in the image. Identify the stitched loop that passes through one of the slots on the Quad/Tri slide.

Note: The strap has two stitched loops. We're only working with the one connected to the quad/tri slide.



Step 3. Cut the stitched loop and remove

Using scissors, carefully cut through the stitched loop, then remove it from the Quad/Tri Slide.

Pre-cut Stitched Loop

Post-cut Stitched Loop

Step 4. Shorten the Strap by at least 12 cm (5”) above the seam.

Cut the strap at least 12 cm (5”) above the seam (to the left in our reference image). The images below demonstrate this cut.

Note: for illustration purposes, the scissors are shown at a much shorter distance than recommended.

Pre-cut Stitched Loop

Post-cut Stitched Loop

Due to variations in strap length between brands, the amount to cut may differ. Remember: you can always cut more off, but you can’t add it back. In our example, a 15 cm (6”) cut was suitable.

Removed Portion

Step 5. Insert the cut end into the open slot making a new loop.

Take the newly cut end and insert it through the now-open slot in the slide. The band should go toward the interior so that when worn, there is no excess strap flopping at the back.

Under normal circumstances, tension should hold this loose end in place. However, it may come loose with repeated use. We leave it to the user to decide if they would like to employ additional methods to secure the band. We propose two feasible options:

1. Create a new seam using a basic sewing kit
2. Use a standard quality stapler with paper staples (preferred method)

If using staples, they should be positioned with the legs initially pointing toward the interior of the band (toward the head area). When properly closed, the staple legs will clinch back outward, facing away from the head. When done properly, staples provide a strong seam, and are virtually invisible and imperceptible to the touch.

Insert the Strap into the Receiver

Now that the strap is prepared, we'll insert it into the receiver.

Step 1. Learn the Squeeze

To insert the strap, you'll need to squeeze the receiver in two directions to expand the width of one of the slots:

- Press down on the back of the receiver on the lateral side of one slot
- Simultaneously, squeeze the top and bottom edges of the receiver to compress it
- If done correctly, the middle section in the back of the receiver will pop out
- Maintain this expansion by continuing to press from the top and bottom

Please refer to the GIF below on how to do this. It is recommended to practice this a few times. Do not be afraid to use adequate pressure.

Step 2. Thread the Strap Through the First Slot

Squeeze the receiver in the manner described above and maintain the slot expansion. Take the end of heart-monitor chest strap without the G Hook, and with the silicone electrodes facing away from the back, thread it through the expanded slot. Thread the strap through until it emerges from the other side with enough length to pull on. Then, pull the strap through until both button connections have passed through.

Step 3. Thread the Strap Through the Second Slot

Thread the strap back through the remaining slot. You can either use your finger to expand the slot or use the squeeze method from Step 1. When finished, the silicone electrodes should be spaced roughly equally when emerging from the back of the receiver, and the button contacts should be equally spaced apart in the inner compartment.

Install the Electronics Case

You're almost there! Two of the three components are now connected. Let's add the final piece: the electronics case.

This section is best performed with a table for support.

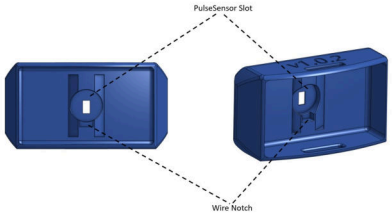
Step 1. Prepare Access to the PulseSensor Slot

To install the PulseSensor, first we need to clear a path to the slot in the receiver. Pull excess strap through the first receiver slot. Loop this strap back over the receiver so it is no longer blocking the front. No part of the strap should now be blocking access.

Refer to the GIF below:

Step 2. Identify the Wire Notch and Orient the Receiver

The PulseSensor slot in the receiver is notched on one side to accommodate the connecting wires. To organize a proper installation, match the version number text on both the receiver and the electronics case so they face upward. When correctly oriented, the PulseSensor's wire notch will be on the bottom.



Step 3. Learn the Squeeze

The PulseSensor shouldn't be forced into the slot. Instead, the slot is meant to expand when force is applied to the back of the receiver, and the inner walls are pulled slightly outward. Practice this squeezing motion before attempting to insert the PulseSensor.

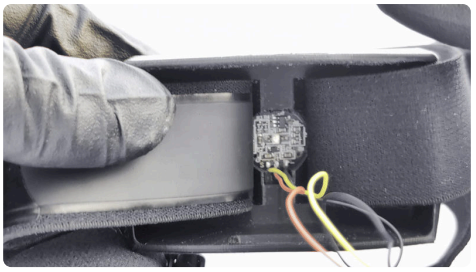
Step 4. Squeeze and Insert the Pulse Sensor

To complete the installation, first lay the electronics case on a table. This way when lifted the PulseSensor will only bear its own weight. Lightly press the PulseSensor into the beginning of the slot, oriented so the wires will sit in the notch.

Now, squeeze the receiver as demonstrated in the previous step. You can use a thumb to hold the PulseSensor in position. As you release the squeeze the receiver should "swallow" the PulseSensor. You may need to apply gentle pressure after for the PulseSensor to completely mate with the back surface of the slot.

Step 5. Unloop the strap

Unloop the strap and return it to its original position.

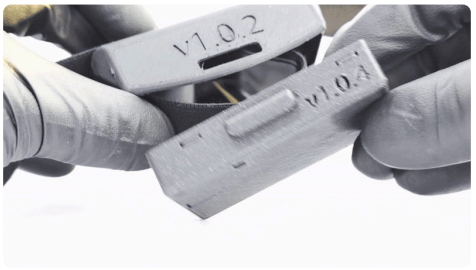


Step 6. Connect the Snap-Fastener Electrodes

Now it's time to mate the electrode connections on the electronics case and the strap.

Ensure that the Version text on the receiver and electronics case are facing the same direction. Button the male snap-fastener on the electronics case to the female snap-fasteners in the strap. There should be an audible click, indicating a snug connection.

If there is no click, or the connection is obviously loose, then the OSSMM will not function.



Step 7. Insert the Electronics Case into the Receiver

Slot the electronics case into the receiver as shown in the GIF below:

Step 8. You've done it!

Reconnect the G Hook and you're done!

You've completed the physical assembly of the OSSMM headband.

Proceed to Step 4. Software (https://jvgiordano.github.io/OSSMM/software/) to complete the required software installations.

Updated: May 16, 2025

Software (https://jvgiordano.github.io/OSSMM/software/)

7 minute read

** Difficulty: ★★☆☆☆ (Beginner/Intermediate) **

Congratulations on assembling your OSSMM headband! Now let's set up the software needed to bring it to life.

The OSSMM system requires two software components:

1. **Arduino Code for Headband** - Manages the sensors, data collection, and Bluetooth communication on the Seeed Xiao nRF52840 Sense microcontroller (MCU)
2. **Android Application** - Software that receives and stores the data Bluetooth stream while providing a user interface (UI) for system control

These instructions are provided using Windows 11, but the process follows the same principles on macOS and Linux.

There is an Advanced section at the end for those interested in modifying the Android App. This is not required!

Arduino Code for the OSSMM Headband

The OSSMM headband is powered by a Seeed Xiao nRF52840 Sense MCU, which performs three critical functions:

1. Regulates power to all components
2. Digitizes analog sensor signals
3. Transmits collected data via Bluetooth Low Energy (BLE)

Before the headband can function, you must install the necessary Arduino code (a C++ variant) on to the MCU.

This section provides step-by-step instructions for setting up the Arduino development environment and uploading the code. If you already have the Arduino IDE installed, you can skip directly to "Install the nRF52 Seeed Board Library" (https://jvgiordano.github.io/OSSMM/software/#install-the-seeed-nrf52-library-and-configure-for-the-xiao-sense-nrf52840-mcu).

Install Arduino IDE

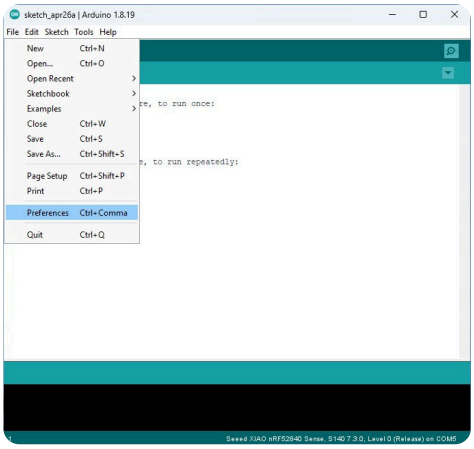
Arduino Integrated Development Environment (IDE) is a code-editor used for working with Arduino Code and associated libraries.

Please install the latest version from the Arduino Website (https://www.arduino.cc/en/software).

Install the Seeed nRF52 Library and Configure for the Xiao Sense nRF52840 MCU

The Arduino IDE requires the Seeed nRF52 Library in order to communicate and compile code for the Xiao Sense MCU. Follow the steps below to install this library and configure the Arduino IDE to be used with the MCU:

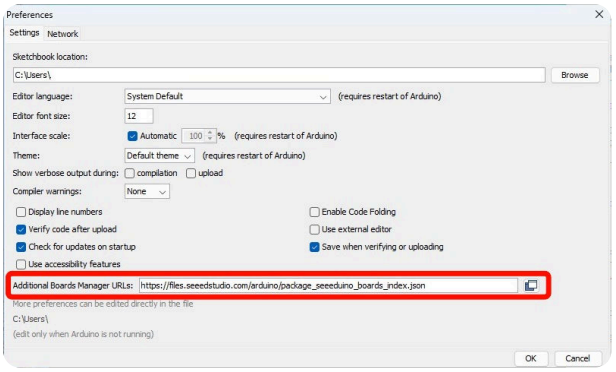
Step 1. With the Arduino IDE open, go to 'File » Preferences'



Step 2. Locate the “Additional Boards Manager URLS” Section and paste in the following:

https://files.seeedstudio.com/arduino/package_seeeduino_boards_index.json

If you already have another Board Manager URL for a different library, use a ‘,’ to separate the URLS.

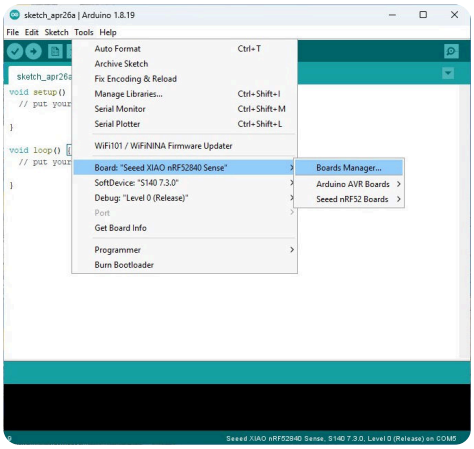


Press “Ok” and exit the preferences window.

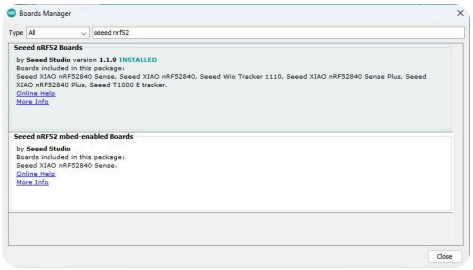
Step 3. Go to ‘Tools » Board: » Board’s Manager’ and install Seeed nRF52 Board Manager.

First go to ‘Tools » Board: » Board’s Manager’ and click on Board’s Manager.

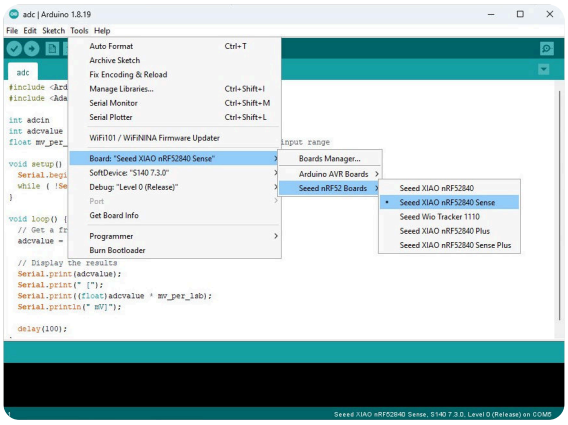
Note: Under Tools, “Board: “ may be empty or contain other names. It does not matter.



In the search bar at the top, type ‘seeed nrf52’. Two board manager options should appear. Install the one that says “Seeed nRF52 Boards.”



Step 4. Select ‘Tools » Board: » Seeed nRF52 Boards » Seeed XIAO nRF52840 Sense’



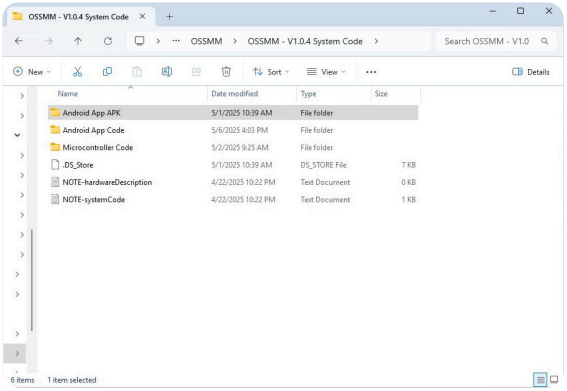
Great! Now the Arduino IDE has the proper software configurations to work with the Seeed nRF52840 Sense. Now the IDE can interact with our board, we can upload the MCU code on to it.

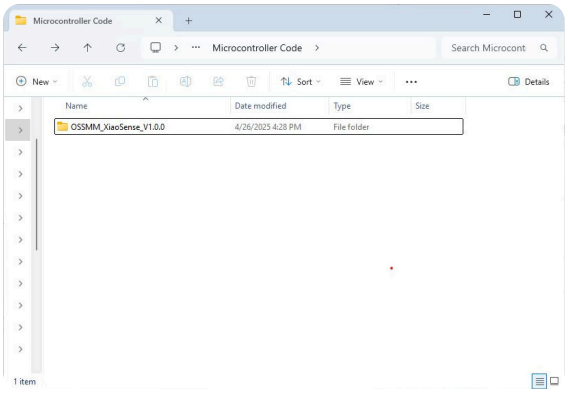
Install the code to the MCU

Now that the Arduino IDE is configured for the the MCU, it’s time to download and install the code:

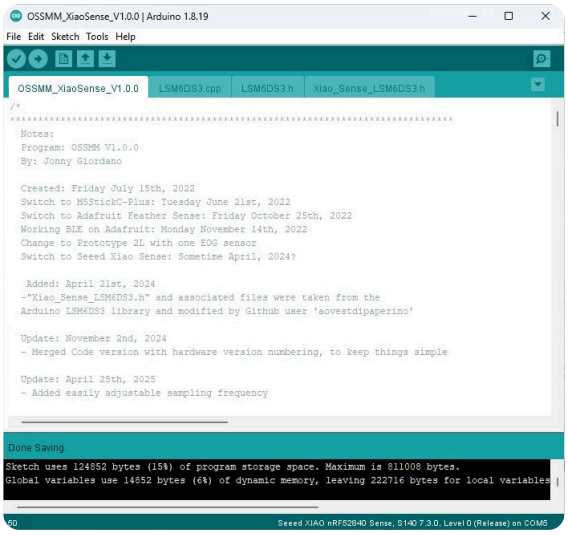
Step 1. Download (or Clone with Git) the Arduino Code from the OSSMM Repository

The code is located under “Microcontroller Code” in the OSSMM - System Code (https://github.com/jvgiordano/OSSMM/tree/main/OSSMM%20-%20V1.0.4%20System%20Code) folder.

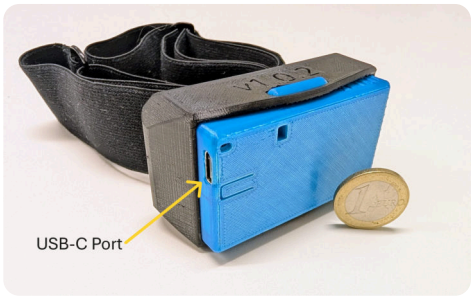




Step 2. Open the Arduino File with the Arduino IDE

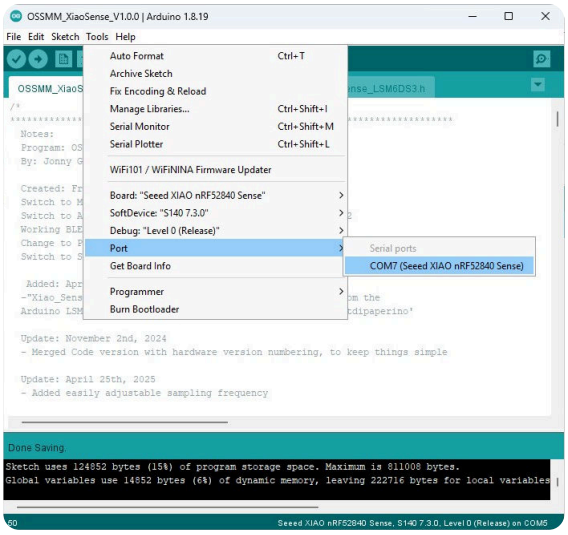


Step 3. Connect the OSSMM Headband to your PC using a USB-C cable



Step 4. Select 'Tools » Port » COM_X_ (Seeed Xiao nRF52840 Sense)'

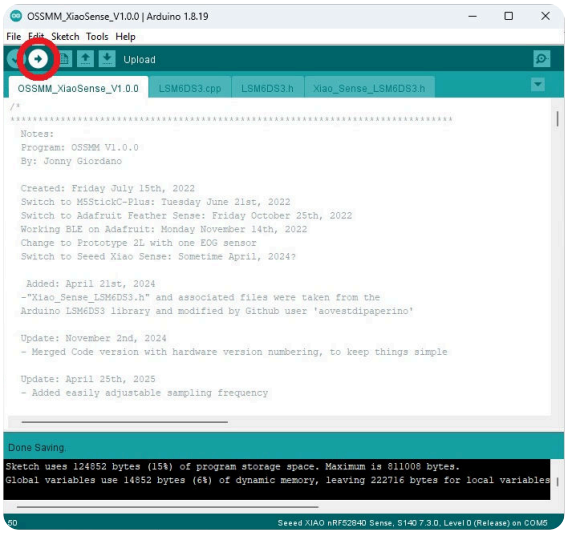
Where COM_X_ will be some port number (e.g., COM7)



This tells the Arduino IDE which port the MCU is connected to. Normally it should automatically detect the port.

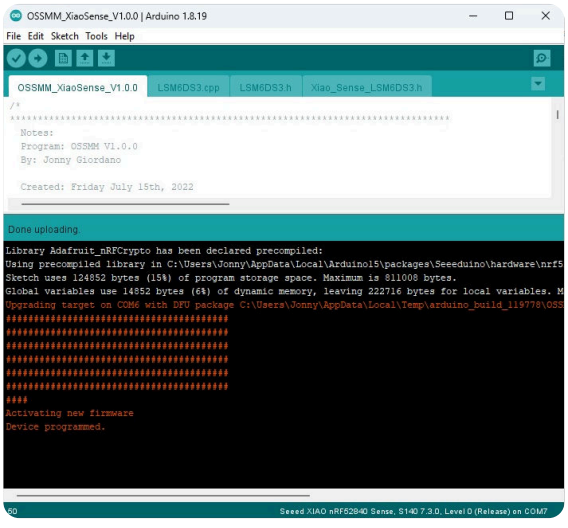
Step 5. Click “Upload”

This will take some time as the code compiles, and then uploads to the MCU.



Step 6. Confirm Upload

Successful code upload will show the following response in the Arduino IDE:



Congratulations! Your OSSMM headband is complete. Now it's time to install the Android companion app!

Install the Android App

The OSSMM system requires a companion app, designed for Android devices, to function. The application performs three critical functions:

1. Provides a User Interface
2. Records data to storage
3. Provides Date-Time information

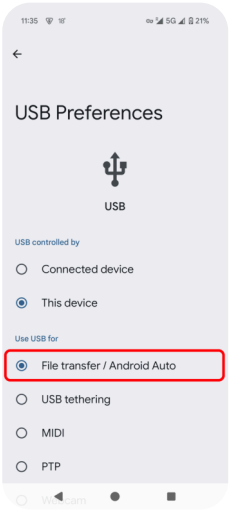
This section provides step-by-step instructions for installing the Arduino apk file onto an Android device. Please note, we used a PC running Windows 11 and a Google Pixel 9 running Android 15 for this demo.

Step 1. Connect your Android Device to a computer via USB with File Transfer Enabled

There are many methods to transfer files to Android devices, but a USB to computer is convenient and fast. We demonstrate this with a Windows 11 machine, but this can be accomplished with Mac OS or Linux. If using Mac OS, you will require a 3rd party File Transfer software like this one (https://apps.apple.com/us/app/macdroid-manager-for-android/id1476545828?mt=12)

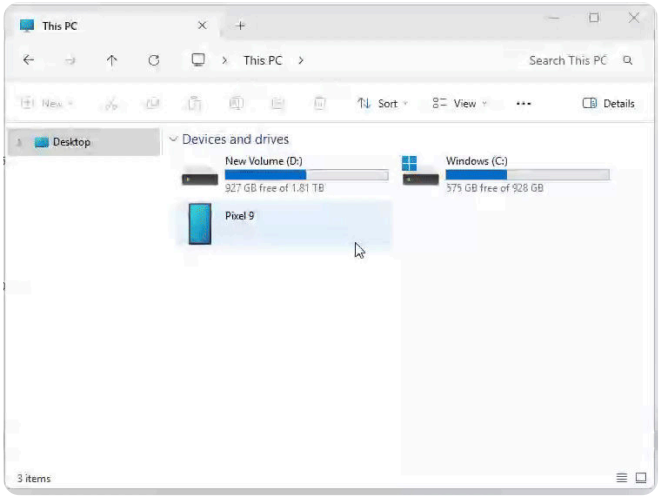
Ensure that "File Transfer" is enabled on your Android device so that the internal memory can be accessed through your computer. You should be automatically prompted to make this selection When you connect the Android device.

Note: If you are not prompted or cannot access the internal memory, your USB cable may only support power transfer and not data transfer. Try with another cable. This may be true even for newer USB-C cables.



Step 2. Access your "Download" folder on your Android Device

Go to "This PC" » Android Device » Internal Shared Storage » "Download"

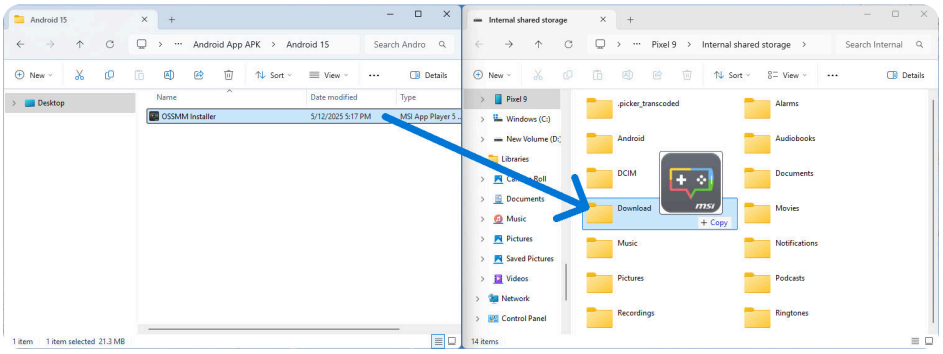


Note: You may choose to use other folders, but we assume the “Download” folder is used here.

Step 3. Move the OSSMM APK to your Android Device

Move the OSSMM APK file from the repository to the “Download” folder on the Android device. You can disconnect the Android device from the computer when the file transfer is complete.

Please use the Android 15 or newer APK file. The Android 9 APK is there for historical posterity. The Android 15 APK contains significant updates, including improved safety features.

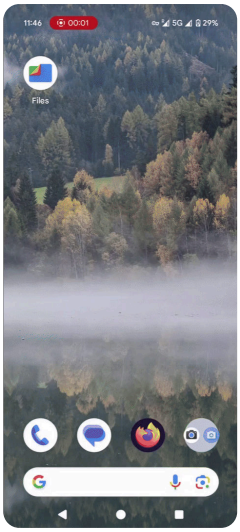


Step 4. Open the “Files” app, navigate to “Download”, and click on the “OSSMM-Installer”

On the Android device, open the “Files” app. This is a standard app for Android and should come with any Android phone released after 2018.

Inside the Files app, navigate to “Downloads”. You will find the “OSSMM-Installer” APK file there. Click on it, then accept the pop-up window which asks if you want to install it.

If you are queried to perform a scan, please do so.



Step 5. You've installed the OSSMM App!

You can now open the OSSMM app!

You will likely encounter a "System Requirements" screen first. Usage of the app requires both Bluetooth and Location settings to be enabled.

Bluetooth is required because the app receives data wirelessly from the OSSMM headband via Bluetooth. Location is needed because it is an Android security requirement for Bluetooth scanning, not because the app actually uses location data. Android requires this because BLE beacons can theoretically be used to determine your location (like in stores or museums), so Google mandates location permission as a privacy protection measure for all apps that scan for Bluetooth devices. The OSSMM headband and app DO NOT collect any kind of location information.

(Advanced Only) Editing Android Application

If you are interested in editing the Android application for your own research, we briefly outline the set-up requirements for doing so. Please note, these are the general steps which must be undertaken and not a step-by-step guide on how to modify the files. Modifying these files requires some familiarity of (or willingness to learn) dart.

Step 1 - Install Android Studio IDE

You can install the latest version of Android Studio here (https://developer.android.com/studio)

Step 2 - Install Flutter SDK

Go to this link (https://docs.flutter.dev/get-started/install) for Flutter Installation.

Step 3 - Install Java

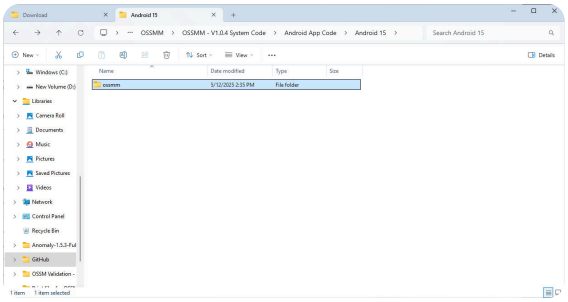
Go to this link (https://www.java.com/en/download/help/download_options.html) for Java Installation instructions.

Step 4 - Configure Flutter and Dart Plugin for Android Studio in Settings

Step 5 - Download (or pull) the OSSMM App Files

These are located under "\OSSMM\OSSMM - V1.0.4 System Code\Android App Code\Android 15" in a

folder called “ossmm”

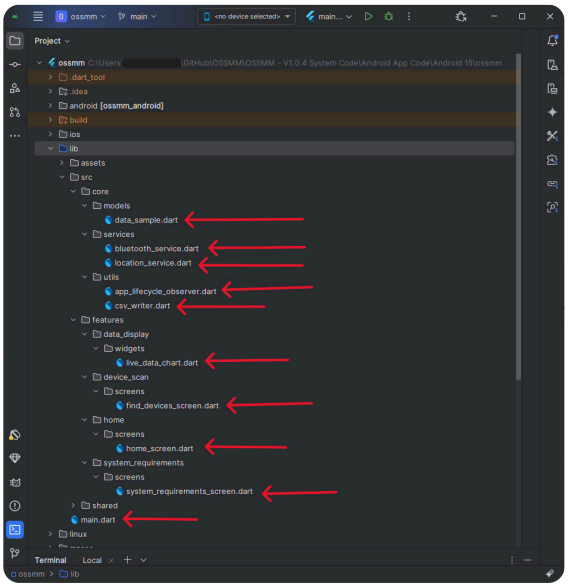


Step 6 - Open the Project with Android Studio

Step 7 - Locate modifiable Android app code within project:

For Android 15 and above, pertinent files are located under the “\ossmm\lib” directory:

Full Directory: “OSSMM\OSSMM - V1.0.4 System Code\Android App Code\Android 15\ossmm\lib”



There are 10 .dart files at the time of publishing (May 13th, 2025). This is likely to expand as more features are developed for the application (e.g., active sleep staging, automated sleep modulation)

Updated: May 16, 2025

Final Checks (https://jvgiordano.github.io/OSSMM/final-checks/)

3 minute read

** Difficulty: ★☆☆☆☆ (Complete Beginner) **

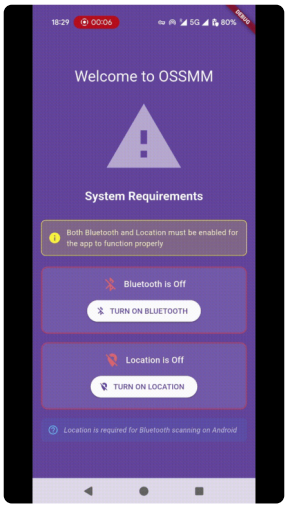
You've assembled the OSSMM system! Now let's verify that it works as intended:

Basic Verification Procedure

Requisites:

1. Android Device with OSSMM App
2. OSSMM Headband
3. Both devices are charged

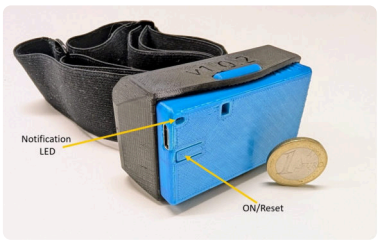
To verify your OSSMM system we will run through the protocol shown in the following GIF. The individual steps are described below:



Basic Verification Demo

Step 1. Turn on the OSSMM headband by pressing the ON/RESET button on the front face.

a. There should be a slight click when pressing the ON/RESET button down. b. Hold the button for 1 second ("One Mississippi"). b. The blue LED will flash several times during start-up.

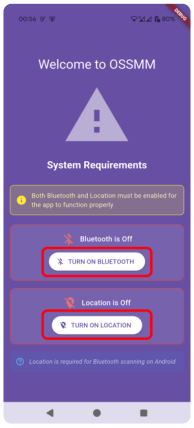


OSSMM Headband ON/Reset and Notification LED placements

Step 2. Put on the OSSMM headband.

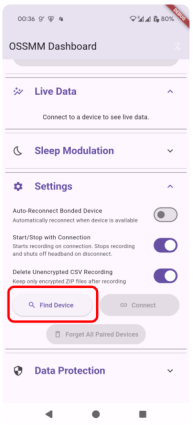
Step 3. Open the OSSMM app on your Android device.

4. Enable the BLE and Location requirements.



System Requirements Screen

Step 5. To bond the app and headband, go to Settings » Find Device.



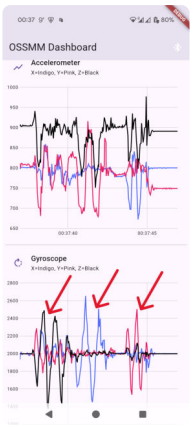
"Find Device" is in the collapsible Settings section.

##Step 6. Connect the OSSMM device. a. The green LED will flash several times for a successful connection.

Bluetooth Devices Scan Screen

Step 7. Move to the Live Data section (this should open automatically)

Step 8. View the Accelerometer and Gyroscope and verify plots show turning of the head.



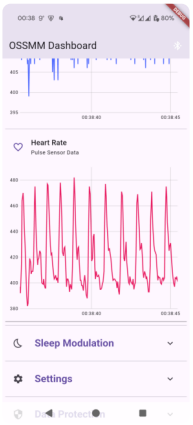
Accelerometer and Gyroscope Plots in the Live Data Section. Arrows point to distinct head turns, hence separate colors in the gyroscope plot.

Step 9. View the EOG section, and verify eye movement is detected with side-to-side saccades or roll your eyes.

In the plot below you can observe left-right saccade, right-left saccade, and clock-wise eye movement.

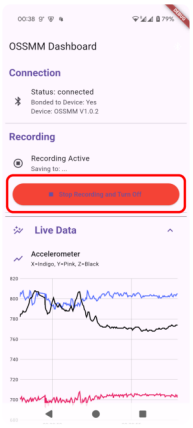
Eye Movement (EOG) plot in Live Data section. Arrows point to distinct eye movements: left-right saccade, right-left saccade, eye rolling.

Step 10. View the Heart Rate plot to see that your pulse is being detected.



Pulse plot in the Live Data section. Each spike shows a heart beat. The EKG 'T Wave' is visible as a low bump after each spike.

Step 11. Press “Stop Recording and Turn Off”



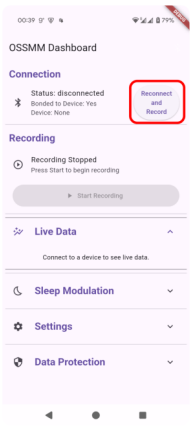
"Stop Recording and Turn Off" button on the OSSMM Dashboard during active Recording.

Bluetooth Bonding Verification

This will confirm the app has bonded to the OSSMM headband. Complete this after “Basic Verification Procedure” and with the same OSSMM headband. BLE pairing is specific to each headband and the “Reconnect” option will only work for that bonded device.

Step 1. Press “Reconnect and Record”

a. The green LED will flash several times for a successful connection.



"Reconnect and Record" button on the OSSMM dashboard. This button is only available after an OSSMM headband has bonded with the app.

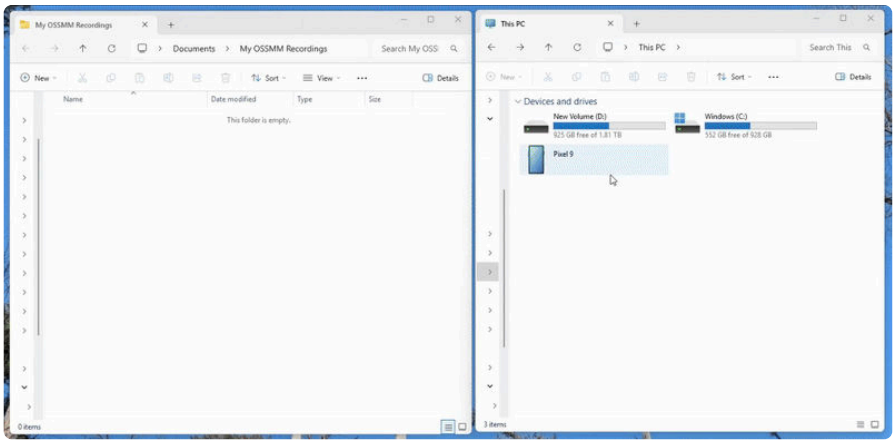
Step 2. Verify the “Live Data” plot function correctly

Step 3. Press “Stop Recording and Turn Off”

Verify Data Saving and Encryption:

The following GIF shows the protocol for verifying data is being saved and encrypted correctly. Individual steps are listed below:

Follow the video below for a demonstration:



Decryption Protocol

Step 1. Connect the Android Device to a computer

Make sure “File Transfer/Android Auto” is enabled.

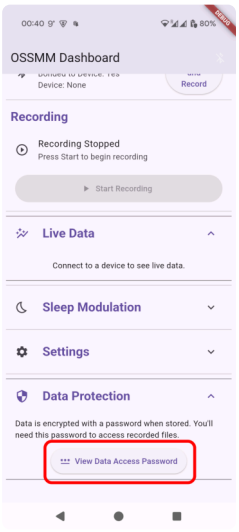
Step 2. Access the internal file system and navigate to “Documents/OSSMM” on the Android device

Step 3. Select the ZIP file corresponding to the time of your verification recording

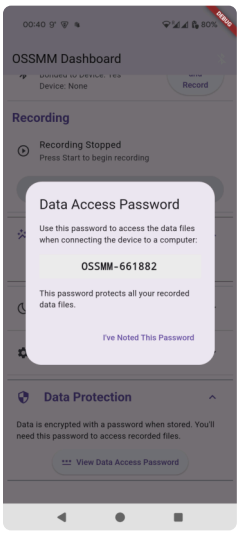
Step 4. Import the ZIP file to your computer, and open with appropriate decompression tool (WinRAR is recommended for Windows)

Note: Standard Windows extraction will not work. Winrar is “free”.

Step 5. Use the password under “Data Protection” » “View Data Access Password” in the app



"View Data Access Password" in the Data Protection Section.



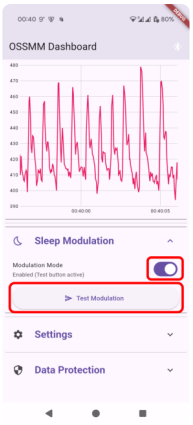
Semi-randomly generated OSSMM password for encrypted recording files.

Note: Each time the application is installed, a new encryption password is semi-randomly generated. This can be modified within the Android app code, so that a chosen password is used instead.

Advanced Verification - if interested in sleep modulation:

(After following the connection/reconnection steps above)

Step 1. Go to “Sleep Modulation” and enable



"Modulation Mode" Toggle and "Test" button in Sleep Modulation section.

Step 2. Press Test Modulation during the connection

- The OSSMM headband should vibrate in a double “blinky” pattern (On-Off...On-Off)

Updated: May 16, 2020

Safety and Data Privacy (https://jvgiordano.github.io/OSSMM/safety-and-data/)

2 minute read

Safety Considerations

User safety was a fundamental priority throughout the development of OSSMM. The design incorporates several important safety features:

- **Low-voltage electronics**: All electronic components are commercially available, hobbyist-grade parts commonly used in wearable maker projects. The system operates entirely on low voltage (3.3V), minimizing electrical risks.
- **Limited battery capacity**: The small-capacity battery (120-220 mAh) significantly reduces potential risks associated with battery malfunctions.
- **Non-invasive sensors**: With the exception of the pulse sensor, all measurement systems are passive. The pulse sensor uses photoplethysmography (PPG) - the same light-based technology found in consumer smartwatches - which emits only low-intensity light to detect blood flow beneath the skin. At no point is current injected into the body.
- **Biocompatible materials**: We selected specific 3D printing filaments based on their published safety data to ensure skin contact compatibility. Safety data sheets for all components and filaments are included in this repository for your reference. You may use your own filaments of choice, but we recommend verifying their respective safety information before usage.

Data Privacy and Security

OSSMM was designed with data protection in mind. The Android 15+ version features:

- **Secure BLE connection**: The app-to-device BLE connection requires verification of 3 unique UUIDs before data transmission occurs. The new version of the app implements Bluetooth bonding, preventing “Man In The Middle” (MITM) attacks during reconnection and ensures only previously authorized devices can pair.
- **User-customizable security**: Each user can modify the UUID values to create their own unique security “profile”.
- **Local data storage**: All data collected by the OSSMM headband is stored locally on a companion Android device in the “Documents/OSSMM” directory. Once data collection is complete, the recorded CSV file is immediately encrypted into a protected ZIP file. By default the unencrypted CSV is deleted, but users may choose to keep the file using a toggle in the settings menu.
- **Device security recommendations**: Due to the nature of sleep and physiological data, we recommend that companion devices used with OSSMM be secured with PIN codes or other access controls.

User Notice

While we’ve made every effort to design a safe system, users assume responsibility for their implementation. We cannot be held liable for any use, misuse, or adverse events resulting from the construction or operation of an OSSMM system. It remains the user’s responsibility to properly assemble their device using appropriate components from reputable sources and to ensure proper operation. This is not a medical device. — *Current Version: V1.0.4*

Updated: May 16, 2025

Performance - Sleep Staging Accuracy (https://jvgiordano.github.io/OSSMM/performance/)

Sleep Staging Accuracy

OSSMM is currently under technical validation for 4-stage sleep classification accuracy.

If you are interested and would like to contribute to this research, let us know!

Updated: April 23, 2025

Demo (https://jvgiordano.github.io/OSSMM/demo/)

less than 1 minute read

This page is a Work-In-Progress.

For a quick demonstration of the app, you can refer to the following GIFs:

- Quick App Demo (https://jvgiordano.github.io/OSSMM/quick-intro/#quick-app-demo)
- Verification Demp (https://jvgiordano.github.io/OSSMM/final-checks/)

Updated: May 17, 2025