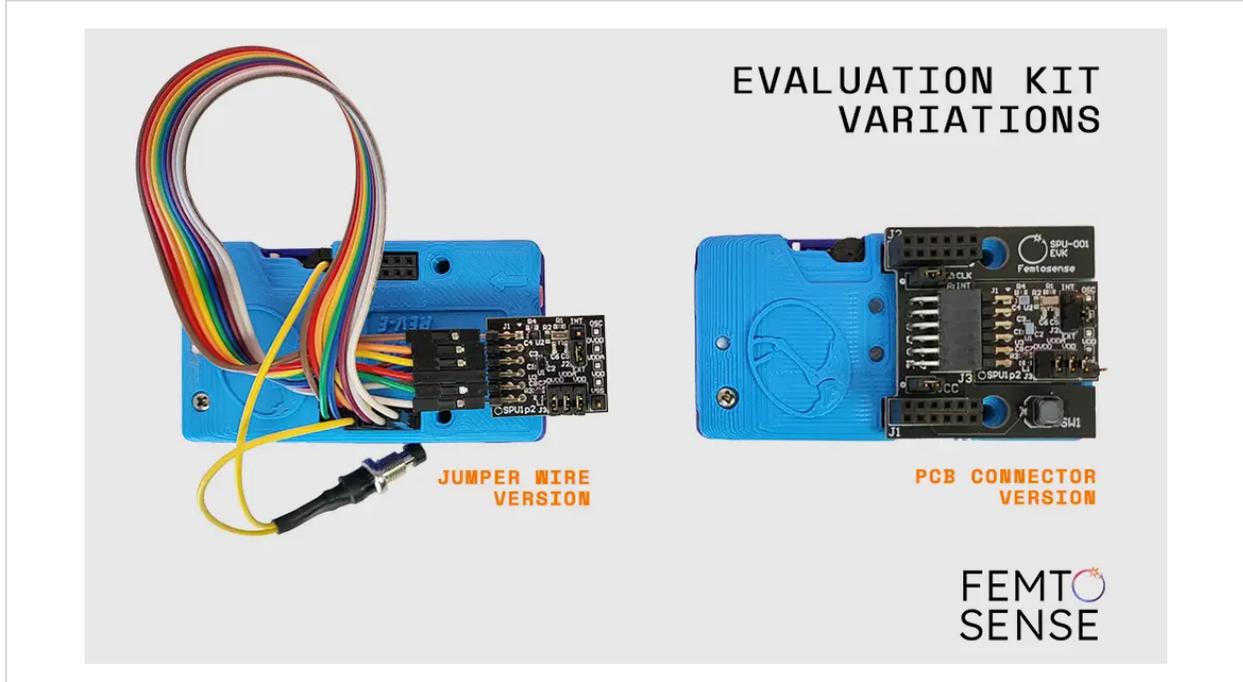


Quick Start Guide v1.6

In your kit, you will find one of the following configurations:

| Jumper Wire Version <i>Release 2023/03/21</i> | PCB Connector Version <i>Release 2023/05/02</i> |
|--|--|
| <ul style="list-style-type: none"> ● SPU-001 circuit board ● Host: Tympan open source hearing aid kit ● USB programming cable ● Ribbon cable of jumper wires for connecting Tympan to SPU-001. ● Button with 2 wires ● Micro SD Card (inserted into the Typman) ● Plastic pencil tool for removing the SD card ● SD to MicroSD adapter | <ul style="list-style-type: none"> ● SPU-001 circuit board ● Host: Tympan open source hearing aid kit ● USB programming cable ● PCB connector board with SPU-001 circuit board (attached to Tympan). Button is built-in with a PCB board. ● Micro SD Card (inserted into the Typman) ● Plastic pencil tool for removing the SD card ● SD to MicroSD adapter |



You may also need:

- SD Card Reader to load new programs into the SD Card
- Headphones with 3.5mm input jack to listen to audio output (e.g. for the AI noise reduction demo)

Table of Contents

- 1. Hardware and Firmware Setup..... 3**
 - 1.1. Setting up..... 3**
 - 1.2. Wire Up the Kit (Jumper Wire Version Only)..... 4**
 - 1.3. Run the Installed Wake Word Detection (WWD) Example..... 5**
 - 1.4. Setting up the Firmware Development Environment..... 6**
 - 1.5. Uploading the Firmware..... 10**
 - 1.6. Running the AI Noise Reduction (AINR) Example..... 10**
 - 1.7. Loading the WWD Example..... 12**
 - 1.8. 0PROG_P Parameters..... 12**
- 2. For Machine Learning Developers..... 14**
 - 2.1 The SPU Development Kit..... 14**
 - 2.2. Setting up the Software Development Kit..... 15**
- 3. Troubleshooting..... 16**
 - 3.1. LED Codes..... 16**
 - 3.2. AINR Example..... 17**
 - 3.3. WWD Example..... 17**
- A. Appendix..... 18**
 - A.1 Femtocrux Windows Setup Guide..... 18**
- Change Log..... 21**

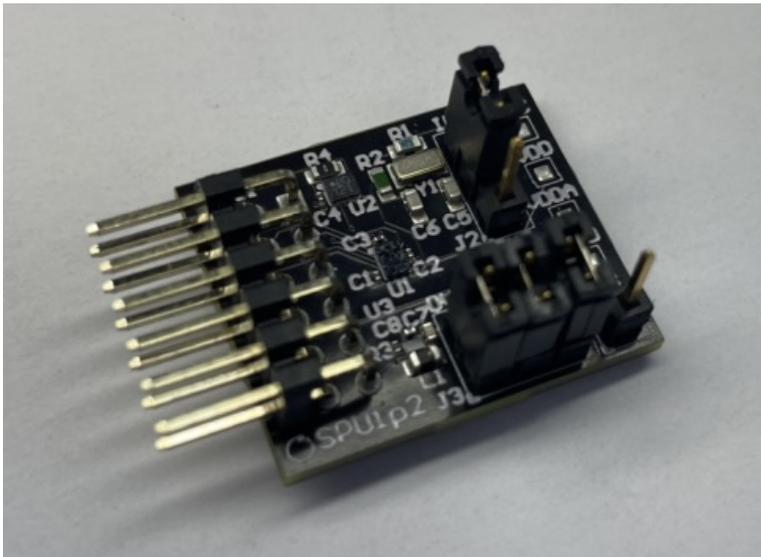
1. Hardware and Firmware Setup

Note: Previous versions of this guide recommended a different jumper and wiring configuration. Please carefully configure your evaluation kit according to this section in order to use the accompanying firmware.

1.1. Setting up

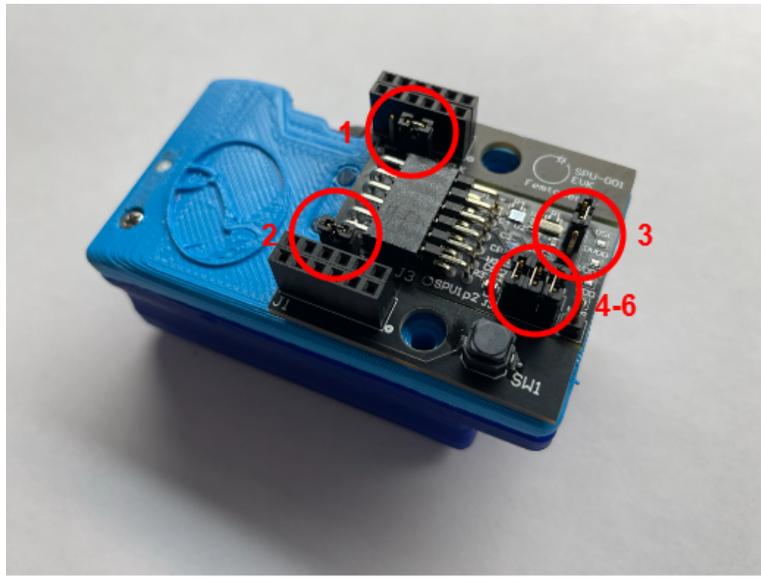
If you have the PCB connector version of the kit, you may skip Section 1.2. If you have the jumper wire version of the kit, please connect it according to the instructions in Section 1.2.

Before continuing, ensure that the 4x 2-pin jumpers on the SPU-001 circuit board are installed. With the 12-pin header facing left as in the photo below, the bottom set of pins, labeled J3, should have 3x jumpers installed vertically. The upper set of pins, labeled J2, should have one jumper installed over the top 2 pins.



Notice the correct placement of the 4x 2-pin jumpers on pin sets J2 and J3 on the SPU-001 circuit board. The component labeled U1 is the WLCSP-packaged SPU-001

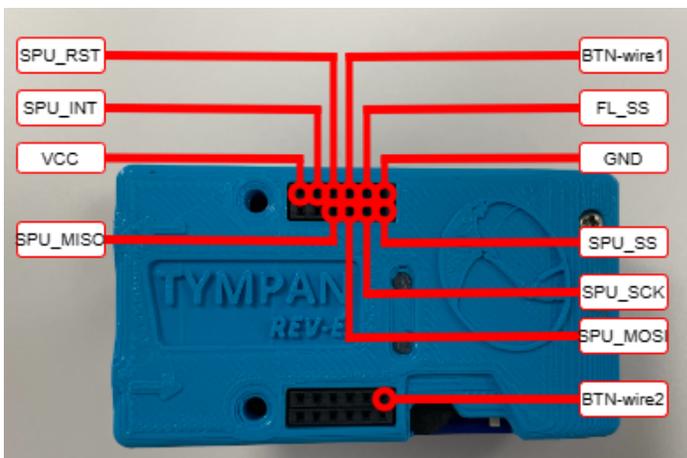
If you have the PCB connector version of the kit, there are two additional jumpers that should be installed, one over the two pins labeled “VCC”, and one over the right two pins labeled “L:CLK R:INT”



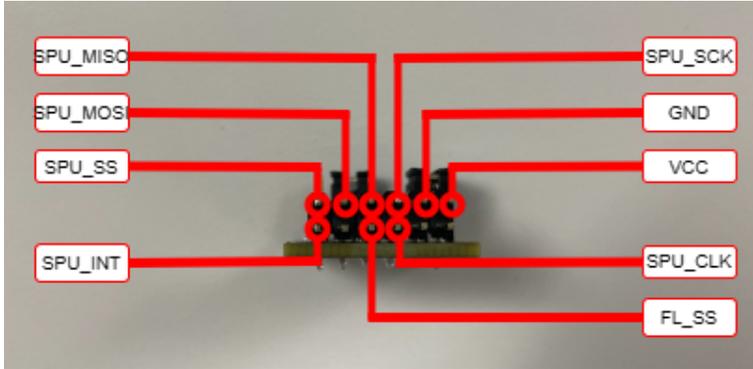
Notice the correct placement of all 6x 2-pin jumpers on the PCB connector version of the kit.

1.2. Wire Up the Kit (Jumper Wire Version Only)

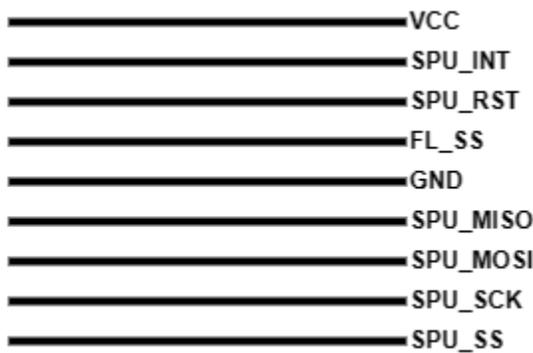
Use the 9-wire ribbon cable to connect the corresponding pins between the host and SPU-001 circuit board. The 2-wire button attaches to the host pins marked BTN-wire1 and BTN-wire2. Note that there is no polarity to the button, and these two wires are interchangeable. Use the following three diagrams to place the wiring. Note that the order of the signals on the ribbon cable is important to minimize noise on the cable.



Wiring diagram of the Tympan host platform (top view). Connect the circled host pins to their corresponding SPU-001 circuit board pins



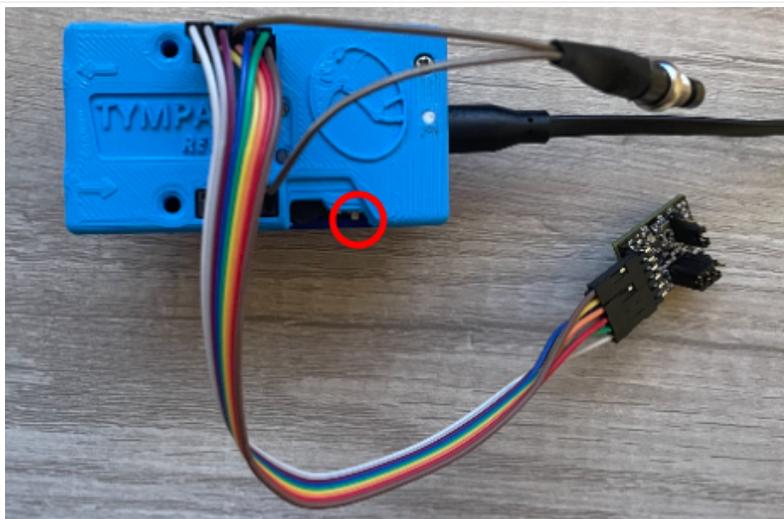
Wiring diagram of the SPU-001 circuit board (side view looking into the 12-pin header). Connect the circled SPU-001 circuit board pins to their corresponding host pins.



The signals on the ribbon cable should be ordered according to the diagram on the left.

1.3. Run the Installed Wake Word Detection (WWD) Example

Once the wiring step above is completed, flip the power switch to the LEFT position to turn it on. If the Tympan does not turn on, the battery may need charging. Plug the Tympan host into a powered USB port and try turning it on again after a few minutes.



Fully wired up EVK (jumper wire version) with the power switch circled in red



EVK (PCB connector version) with the power switch circled in red.

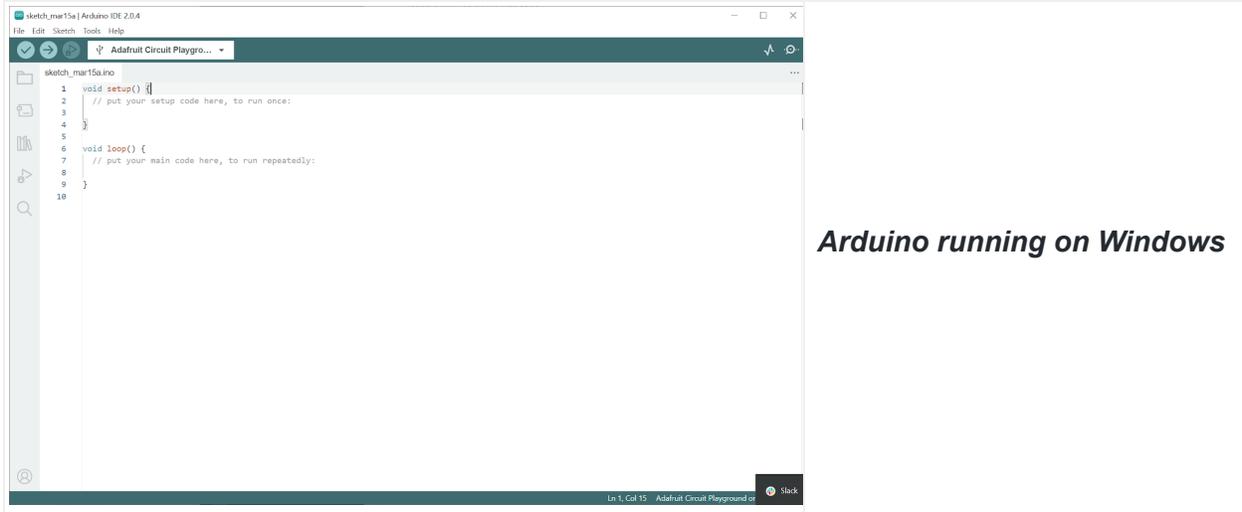
The program will take approximately 10 seconds to boot. After that, you should then see the green LED on top of the host quickly flash 3x. The internal microphone is now listening for the phrase "Hey Snips". When you say this phrase out loud, the green and red LEDs on top of the Tympan should flash¹. If this example is not working for you, see the [Troubleshooting](#) section below.

1.4. Setting up the Firmware Development Environment

In order to change the firmware on the host, you will need to install and configure the Arduino development application. The following instructions are for Windows, but a similar process can be followed for Mac OS or Linux operating systems. However, we have not tested Linux support.

First, install the latest Arduino 2 IDE from: <https://www.arduino.cc/en/software>.

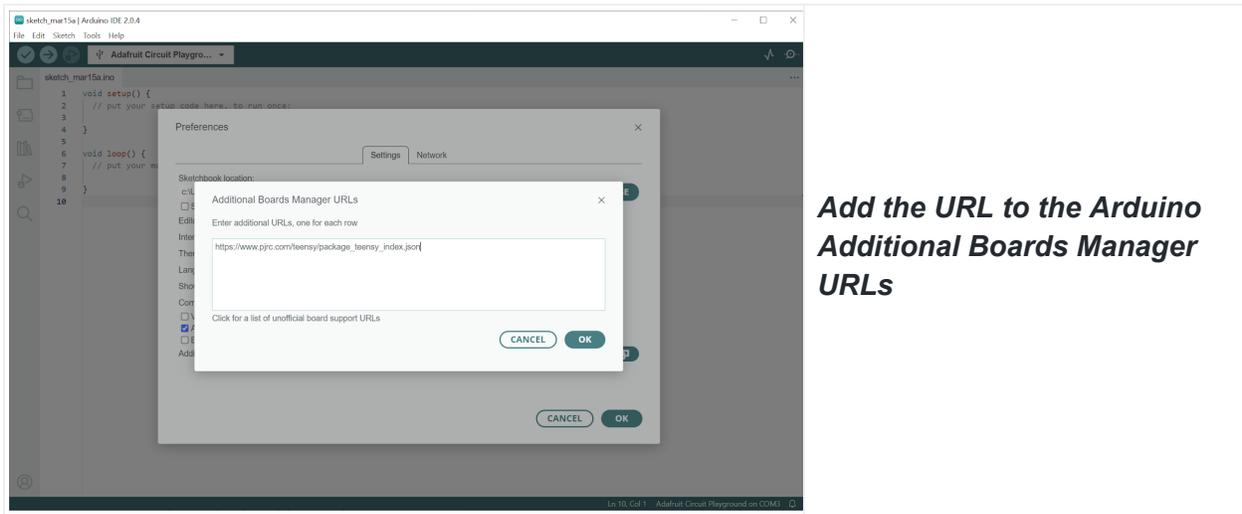
¹ See <https://youtu.be/1VcQGk3lmqs> for a video demonstration.



Arduino running on Windows

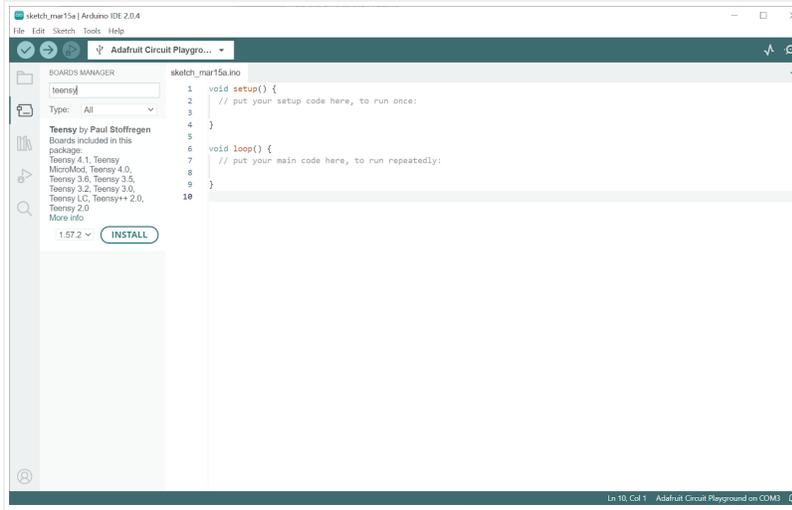
Next, go to **File > Preferences** in Arduino, and click the button next to “Additional board manager URLs”. Add the following line to the URLs:

https://www.pjrc.com/teensy/package_teensy_index.json



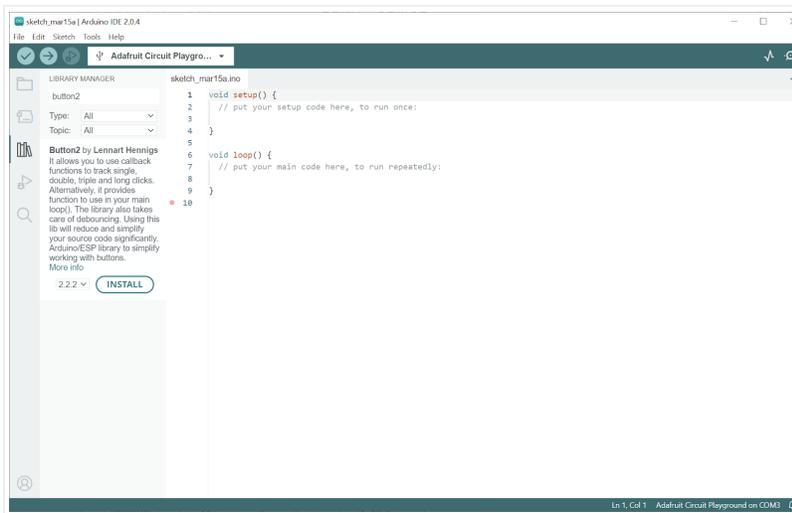
Add the URL to the Arduino Additional Boards Manager URLs

Click OK to go back to the Arduino IDE. Next, click the “Boards Manager” icon on the left side of the IDE and search for “Teensy”. Install the package “Teensy by Paul Stroppfregen”. Choose version 1.57.2 from the dropdown as shown below.



Click to install Teensy

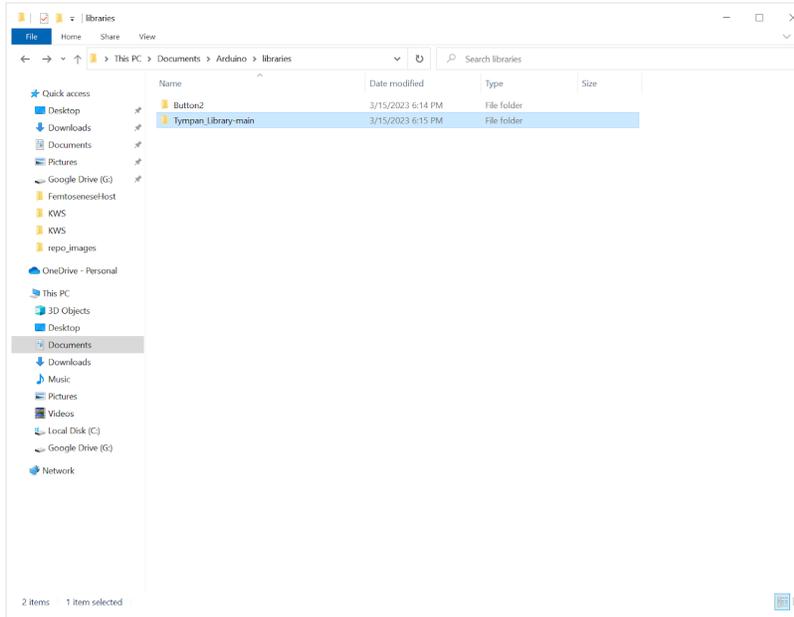
Next, click the “Library Manager” icon on the left side of the IDE and search for “Button2”. Install the package “Button2 by Lennart Hennigs”.



Click to install Button2

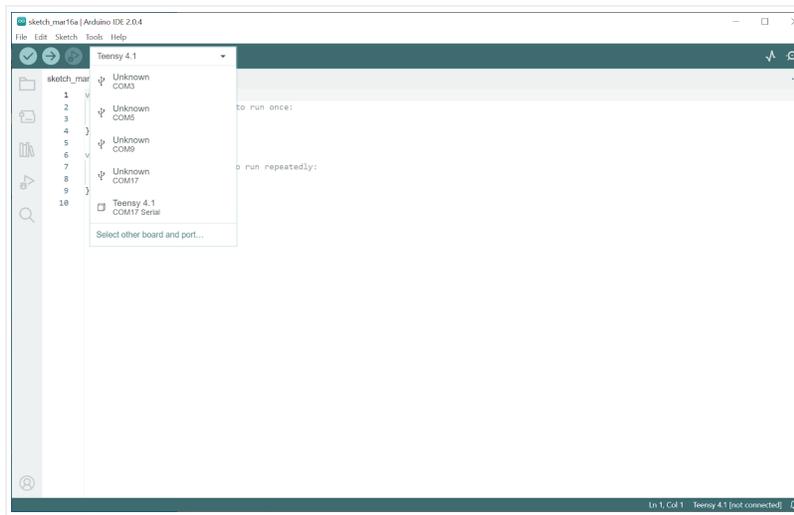
Next, download the Tympan library from Github at: https://github.com/Tympan/Tympan_Library. Download the library by clicking **Code > Download Zip**. Extract the folder to your Arduino Libraries directory. In Windows, this is located at:

C:\Users\<<USERNAME>\Documents\Arduino\libraries



Button2 and the Tympan library should now be in your Arduino libraries directory.

Next, plug in the Tympan host via USB and turn it on. Select **Teensy 4.1** from the board selection dropdown menu at the top of the window.



Select Teensy 4.1 from the dropdown

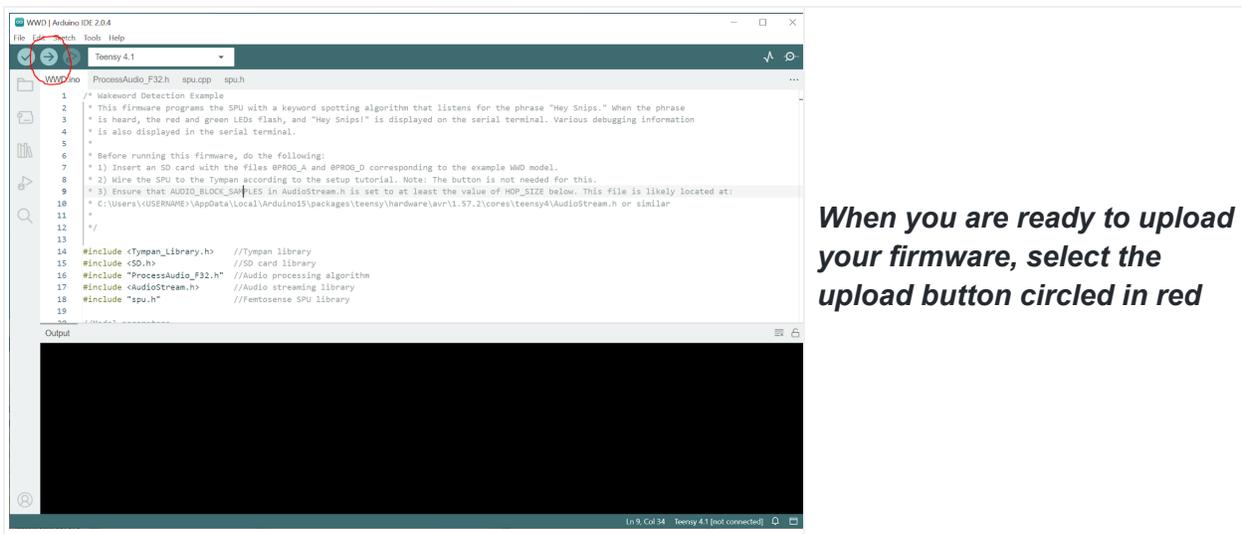
You are now ready to upload firmware from the Femtosense example repository [spu001evk2_model_firmware](#). This code can be provided to you by your Femtosense representative if you do not have it already.

1.5. Uploading the Firmware

To upload firmware, first open the target firmware's `.ino` file in Arduino. Start with the AINR firmware located in:

`spu001evk2_model_firmware/program_files/AINR/AINR.ino`

The `.ino` firmware file includes the program's main functions called `setup()` and `loop()`. It will also open tabs to the supporting code: `spu.h` and `spu.cpp` which include the SPU EVK driver functions, and `ProcessAudio_F32.h`, which includes the audio processing code. When you are ready to upload the code to the Tympan host, plug it in via USB, make sure that "Teensy 4.1" is selected from the dropdown, and click the upload button.



Before booting up the host after upload, make sure to load the `0PROG_A`, `0PROG_D`, and `0PROG_P` files for your model onto the SD card. These files are located in:

`spu001evk2_model_firmware/program_files/AINR/AINRGP_16khz_4hop_8algo_v3`

1.6. Running the AI Noise Reduction (AINR) Example

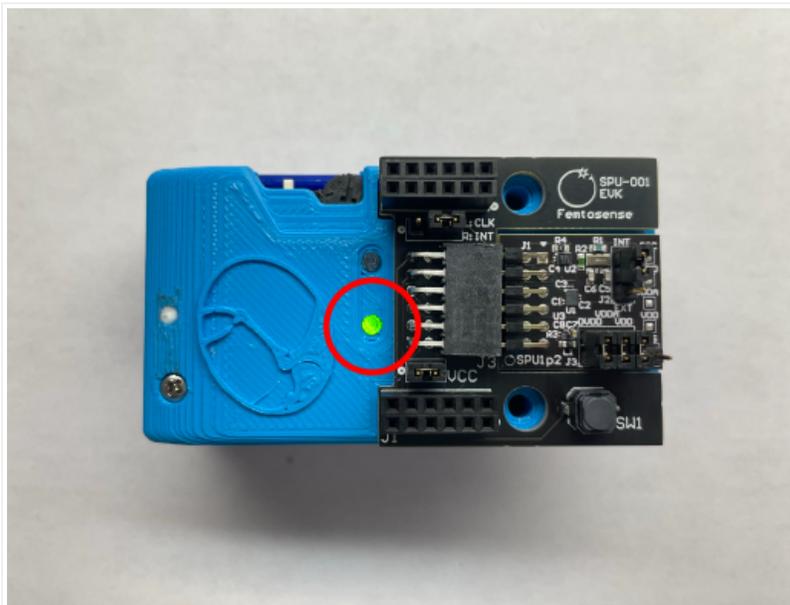
Once the code is uploaded, and the 3 programming files are loaded onto the SD card, flip the power switch to the RIGHT position to turn it on. The program will take approximately 10 seconds to boot. You should then see the green LED on top of the host quickly flash 3x.

Plug in your headphones to the BLACK audio jack on the Tympan host. You should hear loopback audio (unprocessed) from the microphone.



The black and pink 3.5mm jacks are for audio output/input. The example shipped with the EVK uses the internal microphone by default. To change the mic input to use an external mic, set `USE_EXTERNAL_MIC` in the parameters file as shown in the [OPROG_P parameters table](#).

You can toggle the AINR processing on and off using the button marked SW1 on the black PCB. When AINR processing is enabled, the green LED will illuminate and you should hear human speech sound, while background noise is significantly reduced. Output volume on the headphones can be adjusted via the knob next to the power switch. If this example is not working for you, see the [Troubleshooting](#) section below.



The green LED illuminates during AINR processing.

1.7. Loading the WWD Example

To revert back to the WWD example, load the WWD firmware located in:

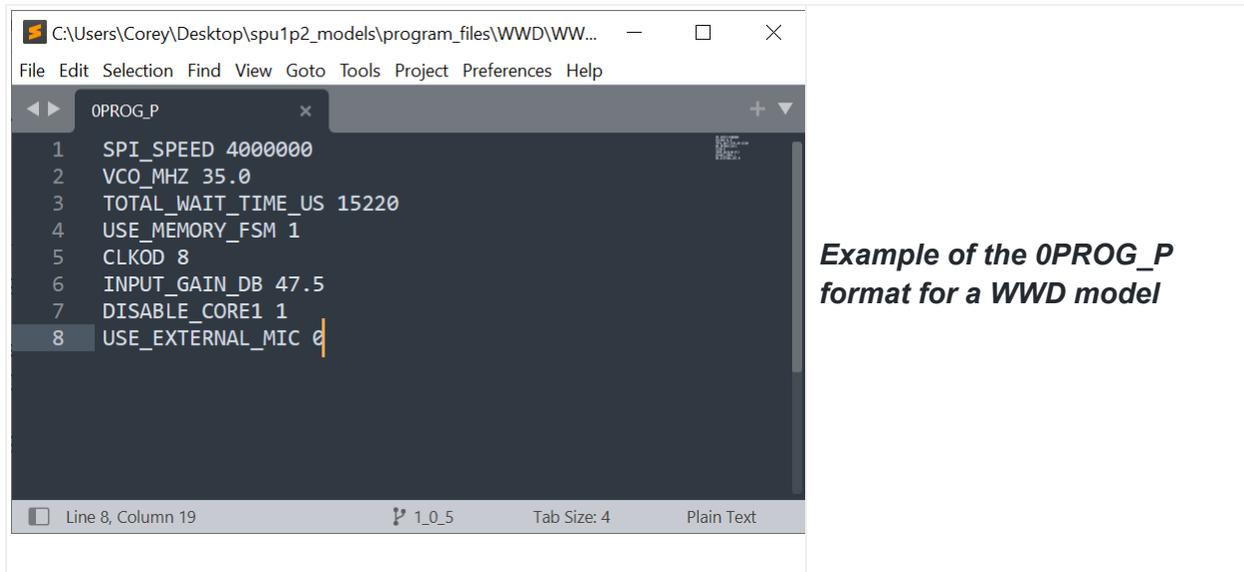
```
spu001evk2_model_firmware/program_files/WWD/WWD.ino
```

Before booting up the host after upload, make sure to load the `0PROG_A`, `0PROG_D`, and `0PROG_P` files for your model onto the SD card. These files are located in:

```
spu001evk2_model_firmware/program_files/WWD/WWDSNIPS_8khz_16ms_v2
```

1.8. 0PROG_P Parameters

Several parameters can be used to adjust the firmware, which can be loaded into the `0PROG_P`. The format is shown in the following screenshot:



The following table describes these parameters:

| Parameter | Description | AINR Support | WWD Support | Min | Default AINR | Default WWD | Max |
|---------------------------|---|--------------|-------------|---------|--------------|-------------|----------|
| SPI_SPEED | SPI Speed between SPU and Tympan (Hz) | Yes | Yes | 4000000 | 4000000 | 4000000 | 50000000 |
| VCO_MHZ | SPU Clock Speed (Mhz) | Yes | Yes | 30 | 100 | 100 | 200 |
| CORE_GATING_EN | Enable Core Clock Gating (0=off, 1=on) | Yes | No | 0 | 0 | - | 1 |
| CORE0_READY_US | Core Clock Gating Parameter (us) | Yes | No | 0 | 0 | - | - |
| CORE_TRANSFER_US | Core Clock Gating Parameter (us) | Yes | No | 0 | 0 | - | - |
| SAFETY_MARGIN_US | Core Clock Gating Parameter (us) | Yes | No | 0 | 0 | - | - |
| TOTAL_WAIT_TIME_US | Total wait time before checking for SPU interrupt (us) | Yes | Yes | 0 | 0 | 0 | - |
| USE_MEMORY_FSM | Enable memory power-saving mode on SPU (0 or 1) | Yes | Yes | 0 | 0 | 0 | 1 |
| CLKOD | Default Clock Divider with respect to VCO_MHZ. In AINR, the clock is divided during non-processing time (during SPI I/O). In WWD, the divider is always applied. Must be 1 or even. | Yes | Yes | 0 | 1 | 1 | 16 |
| TUNE_DEBUG | Turn on Serial printing of tuning and extra debug data (0=off, 1=on) | Yes | Yes | 0 | 0 | 0 | 1 |
| DISABLE_CORE1 | Turn off second core if not needed (0=core1 enabled, 1=core1 disabled) | No | Yes | 0 | - | 0 | 1 |
| USE_EXTERNAL_MIC | Use external mic (0=internal mic, 1=external mic) | Yes | Yes | 0 | 0 | 0 | 1 |
| OUTPUT_MAX_GAIN_DB | Maximum output gain on audio output jack (dB) | Yes | No | -20.0 | 0 | - | 40.0 |
| INPUT_GAIN_DB | Input gain from microphone input to SPU (dB, 0.5dB steps) | Yes | Yes | 0 | 25.0 | 47.5 | 47.5 |

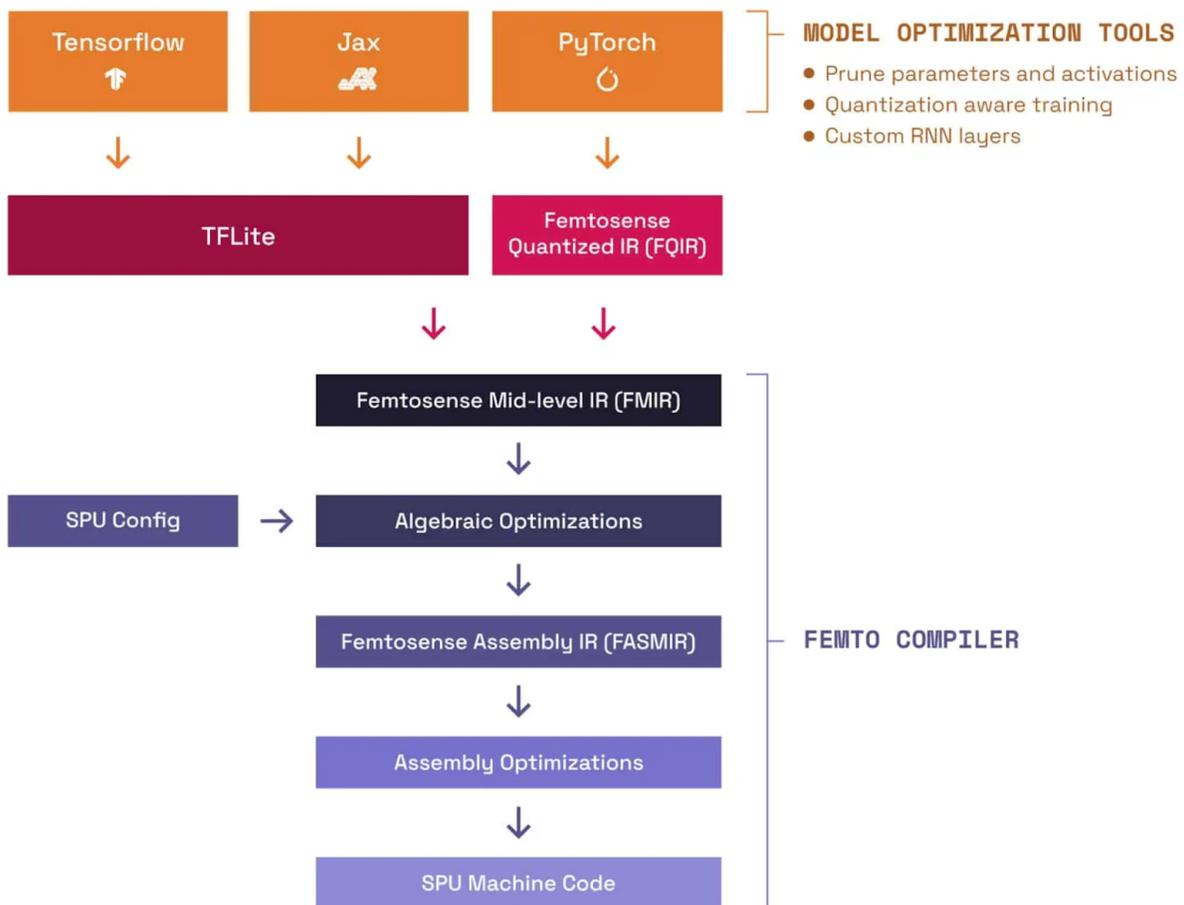
Note that timing parameters (all parameters except those in the last 3 rows of the table above) should not be adjusted unless necessary. The models provided by Femtosense include OPROG_P files with good parameters for power optimization.

2. For Machine Learning Developers

2.1 The SPU Development Kit

Femtosome provides a SPU-Development Kit (SPUDKIT) to help users compress and deploy their own models to the EVK or SPU.

Users can use our packages to go from PyTorch or TensorFlow and produce a serialized format of their quantized models. From there, they can use our compiler to produce deployable binaries. The user flow is represented by the graph below:



Note that these tools are not necessary to run the demos above but rather for developing and deploying your own model to the EVK.

2.2. Setting up the Software Development Kit

FemtoseNSE provides the Python software packages listed below for model development and deployment.

| Package | Source | Description | Documentation ² |
|--------------------------------|--------|--|---|
| fmot | PyPI | The PyTorch frontend for FemtoseNSE | https://fmot.femtoseNSE.ai/ |
| femtoflow | PyPI | The Tensorflow frontend for FemtoseNSE | https://femtoflow.femtoseNSE.ai/ |
| femtocrux | PyPI | The FemtoseNSE compiler | https://femtocrux.femtoseNSE.ai/ |
| femtodriverpub | Github | Utilities to package up compiler output for firmware | See README in github repository |

As prerequisite to installation, you will need the following on your system

- Python version 3.10 or later
- [Docker](#)

Make sure these are installed before installing the FemtoseNSE packages.

FemtoseNSE packages are hosted on the PyPI and can be installed with `pip`. For example,

```
Unset
pip install fmot femtoflow femtocrux
```

Note that when you run `femtocrux` for the first time (or after an update), you will be prompted for a password to download the docker image containing the compiler internals. Contact your FemtoseNSE representative if you do not have or cannot find your password.

`femtodriverpub` is cloned directly from github.

To deploy a custom model to the EVK, we recommend starting with the [femtocrux documentation](#).

² Use your FemtoseNSE-provided password to access the documentation. Contact your FemtoseNSE representative if you cannot find or do not have a password.

3. Troubleshooting

3.1. LED Codes

The example code includes several error codes to help you debug your EVK. Errors will be displayed as a flashing red LED as follows:

| LED Code | Description |
|----------------|--|
| green 3x | Bootup was normal |
| red 1x | This non-blocking error indicates that the <code>0PROG_P</code> parameters file was not found. Default parameters will be used instead. |
| red 2x | This error indicates that the <code>AUDIO_BLOCK_SAMPLES</code> definition in your <code>AudioStream.h</code> file is too low. It should be increased to at least the value of <code>HOP_SIZE</code> . This file is located at: <code>C:\Users\<USERNAME>\AppData\Local\Arduino15\packages\teensy\hardware\avr\1.57.2\cores\teensy4\AudioStream.h</code> or similar in the Windows operating system. |
| red 3x | This indicates that the SPU integrity check has failed. Check the connection between the SPU-001 circuit board and the Tympan host. Also, check that all jumpers are correctly installed. |
| red 4x | This indicates that the programming files cannot be read from the SD card. Make sure that the SD card is inserted and that the two required programming files (<code>0PROG_A</code> and <code>0PROG_D</code>) are in the root directory. |
| red 5x | This indicates that the SPU is not able to receive an input clock signal. Check that the clock jumpers on the SPU are in the correct configuration, and that the clock signal is active if using an external clock. |
| red 6x | This indicates that the SPU has malfunctioned during processing. If using core power gating, check that timings are correct. Also check that the PLL clock speed is high enough. This error will also occur if you touch the crystal and nearby capacitors on the PCB with your finger. If this error occurs immediately after bootup, check that the model files on the SD card are correct. |
| Constant green | This indicates that the SPU is taking too long to process an audio frame. The button will no longer function, and the LED will constantly illuminate green. Increase the PLL clock speed so that the SPU will process the audio frame faster. |

More debug information can be gleaned by connecting a serial terminal to the Tympan host's USB com port at 115200 baud (8-N-1). If the hardware and firmware report normal operation, review the troubleshooting notes below.

3.2. AINR Example

Objective

The output audio should preserve human speech while removing background noise.

Troubleshooting

You should expect the algorithm to perform well in positive SNR noise conditions.

If you are experiencing distortions of speech at 0+ dB SNR, make sure that your testing environment is not too reverberant. Adjust the input SNR accordingly to balance the additional background noise caused by reverberation. The input gain of the microphone can be adjusted using the `INPUT_GAIN_DB` definition in `0PROG_P`. The maximum value for this parameter is 47.5, and should be set in 0.5 dB steps.

If the output is too quiet, check the output volume on the headphones and on the Tympan via the knob next to the power switch.

You may also set the value of `OUTPUT_MAX_GAIN_DB` in `0PROG_P` in order to set the max headphone volume level. The maximum value for this parameter is 40.

3.3. WWD Example

Objective

The light bulb on the Tympan should blink when you pronounce `Hey Snips`, showing that the wake word has been detected.

Caveat

As the model has been provided as a proof of concept, it has only been trained and tested on a dataset of the wake word spoken with an American accent. The performance might degrade with different accents.

Troubleshooting

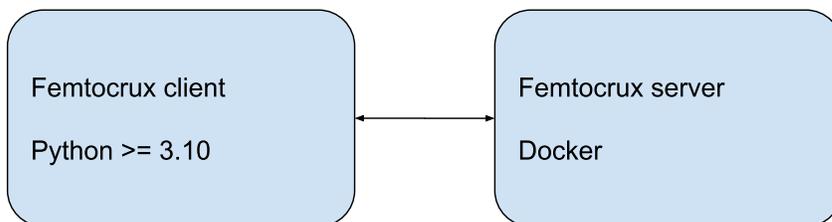
If the wakeword is not detected in a silent environment in a close-up situation (talking in front of the Tympan), test the model by playing some of the test samples with American accents that we have added [here](#).

The provided firmware sets the microphone gain level to a good initial value. Still, you can adjust the firmware to change microphone gain if desired by adjusting the `INPUT_GAIN_DB` parameter in `0PROG_P`.

A. Appendix

A.1 Femtocrux Windows Setup Guide

Note: Femtocrux has been tested on Linux and Window systems. The following guide is for the Windows operating system, but similar steps apply to other systems as well. See [here](#) for the recommended way to install Docker on your system.



This guide explains how to install Femtosense’s Femtocrux compiler on the Windows operating system. There are three main steps.

1. Install Docker, and configure for Linux containers
2. Install Python 3.10
3. Install Femtocrux client and pull Docker image

Install and configure Docker Desktop

Femtocrux’s backend runs in a Docker Linux container. To run it on a Windows machine, we first need to install Docker Desktop, then configure it to run Linux containers.

Install Docker Desktop

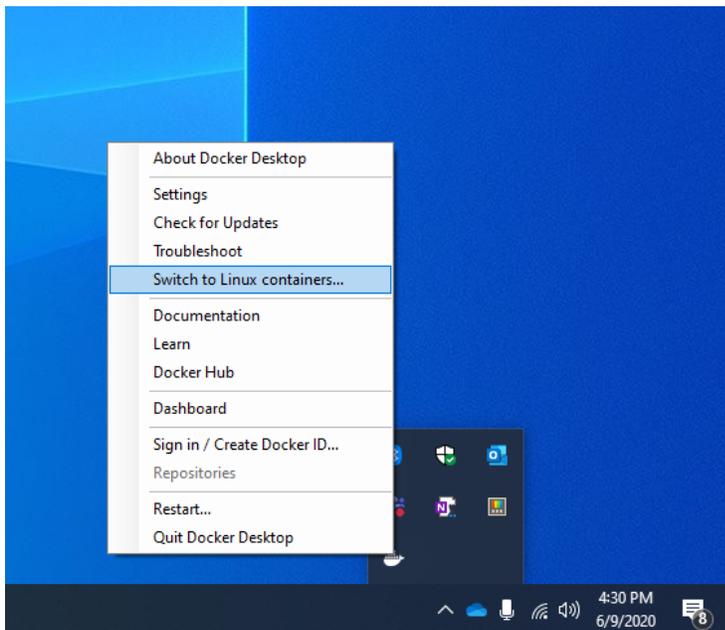
First, install Docker Desktop by following these [instructions](#).

- Download the Docker Desktop installer.
- Click through and install.
 - Docker supports two different backends for Windows: WSL and Hyper-V. Some systems only support one or the other. Although both should work, our internal testing uses Hyper-V.

Configure Docker to use Linux containers

Right click on Docker Desktop, and select “Run as Administrator.” Once Docker Desktop starts, you should see a small whale icon in the taskbar in the lower right corner of your screen.

Next, configure Docker to use Linux containers by right-clicking on the whale icon. See this [guide](#) for more info.



Install Python 3.10

Femtocrux's client requires Python (version ≥ 3.10) to run. The installer can be downloaded [here](#). For most systems, we recommend choosing "Windows Installer (64-bit)." Simply download the installer and click through it, choosing to add Python to your system's PATH.

To check that your Python installation is discoverable from the command line, run the command:

```
C:\Users\Administrator>python --version
Python 3.10.11
```

Install Femtocrux

The following commands install the Femtocrux client and server.

Install Femtocrux client

The Femtocrux client is available on [pypi](#). To install the latest version, run the following command.

```
Unset
```

```
python -m pip install femtocruX
```

Install FemtocruX server docker image

To check that FemtocruX works, you can try running the following command.

```
Unset
```

```
python -c "from femtocruX import CompilerClient; CompilerClient()"
```

If this is your first time running this version of FemtocruX, you will be prompted to log in to Github Container Registry and pull the Docker image.

```
C:\Users\Administrator>python -c "from femtocruX import CompilerClient; CompilerClient()"
Failed to find the docker image ghcr.io/femtosenX/femtocruX:0.2.8-1 locally.

    Attempting to pull docker image from remote.

    Alternatively, you can pull the image yourself with the command:
        docker pull ghcr.io/femtosenX/femtocruX:0.2.8-1

Please enter your Femtosense-provided key: _
```

At this point, please copy and paste the password provided to you by Femtosense, and press enter. If authentication succeeds, the client will start pulling the FemtocruX Docker image. This may take a few minutes to complete.

Once the download completes, you can run the same command to verify that the server is working.

```
C:\Users\Administrator>python -c "from femtocruX import CompilerClient; CompilerClient()"
Checking container status...
Container started successfully.
Container passed health check.
Created gRPC channel at 172.22.6.19:50051
waiting to establish a connection...
Connection successful.
```

Change Log

| Version | Release Date | Description |
|---------|--------------|---|
| 1.0 | 2023-06-16 | Initial release |
| 1.1 | 2023-04-05 | Changes related to loading 8ms AINR models |
| 1.2 | 2023-04-13 | Added note about Teensy version and ribbon cable arrangement |
| 1.3 | 2023-04-14 | Typo in ribbon cable diagram |
| 1.4 | 2023-05-03 | More documentation about the the SDK, added new error codes, added info on PCB connector board for Tympan |
| 1.4s | 2023-05-04 | WWD and 16ms model info removed, jumper wire info removed, new error codes added. |
| 1.5s | 2023-06-13 | Add info about 0PROG_P file and auto mic detect |
| 1.6s | 2023-06-16 | Added WWD info back, removed non-HW content |
| 1.6 | 2023-07-23 | Added section 1.5 - guide to update new AINR firmware Added section 1.7 - guide to revert to WWD firmware Added section 1.8 - 0PROG_P parameter table Added Appendix - Femtocrux Windows Setup Guide |