

# Quick Start Guide v2.0

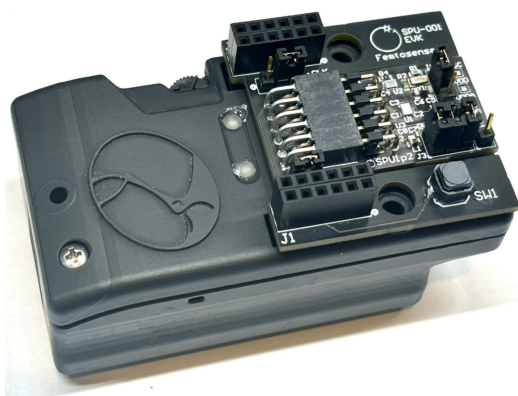
In your kit, you will find the following:

- EVK2 assembly consisting of:
  - SPU-001 Evaluation Board (“EVB”) that contains the WLCSP-packaged SPU-001 processor
  - Tympan open source hearing aid host
  - PCB connector board with buttons connecting the EVB to the Tympan host.
  - Micro SD Card (inserted into the Tympan)
- USB programming cable
- Plastic pencil tool for removing the SD card and pressing the small program button
- SD to MicroSD adapter

Depending on the release date, your EVK2 assembly matches one of the following photos. The primary difference between these two is that EVK2v2 contains two buttons, and uses the SPU-001 mass-production chip, whereas EVK2v1 contains one button and the SPU-001 test chip. These kits are mostly backwards compatible.

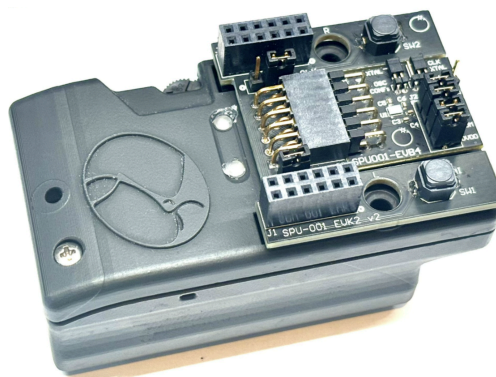
## EVK2v1 assembly

*Release 2023/05/02*



## EVK2v2 assembly

*Release 2024/03/05*



A previous version of EVK2 was also released that used rainbow jumper wires between the Tympan Host and the EVB. This version is still supported, but please contact Femtosense if you would like to upgrade your jumper wire version to one of the versions above.

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)

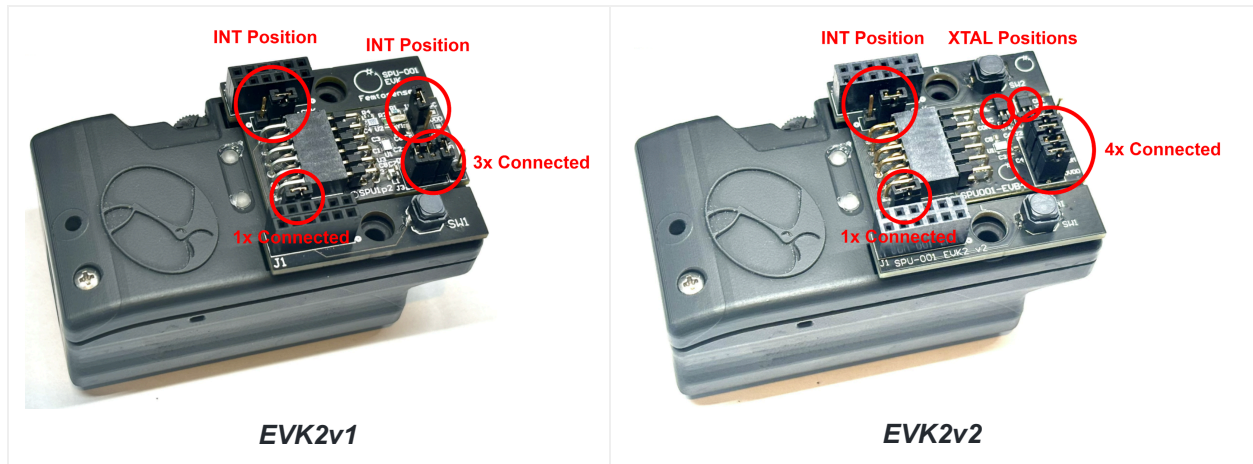
# Table of Contents

|   |    |
|---|----|
| 1. Hardware and Firmware Setup.....                       | 3  |
| 1.1. Setting up.....                                      | 3  |
| 1.2. Run the built in example.....                        | 4  |
| 1.2.1. AI-Noise Reduction (AINR) demo.....                | 5  |
| 1.2.2. Speech Commands Demo.....                          | 5  |
| 1.3. Changing the model and/or firmware.....              | 6  |
| 1.4. Storing Multiple Models on the SD card.....          | 8  |
| 1.5. XPROG_P Parameters.....                              | 8  |
| 1.6. Setting up the Firmware Development Environment..... | 11 |
| 1.7. Uploading the Firmware from Arduino.....             | 14 |
| 2. For Machine Learning Developers.....                   | 15 |
| 2.1 The SPU Development Kit.....                          | 15 |
| 2.2. Setting up the Software Development Kit.....         | 17 |
| 3. Troubleshooting.....                                   | 18 |
| 3.1. LED Codes.....                                       | 18 |
| 3.2. AINR Examples.....                                   | 19 |
| 3.3. Speech Command Examples.....                         | 19 |
| A. Appendix.....  | 20 |
| A.1 Femtocrux Windows Setup Guide.....                    | 20 |
| Change Log.....   | 23 |

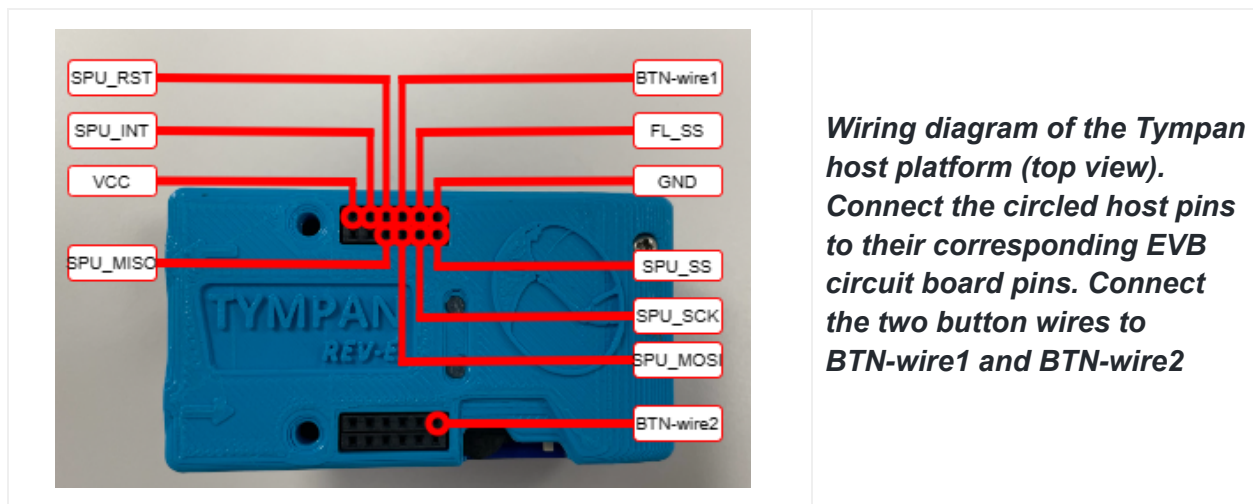
# 1. Hardware and Firmware Setup

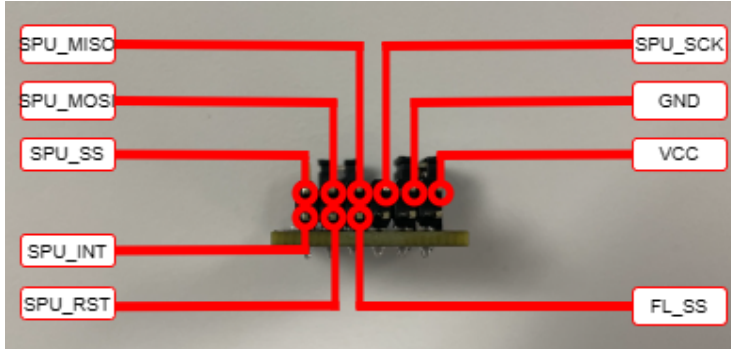
## 1.1. Setting up

Before continuing, ensure that the jumpers are configured correctly as in the photos below

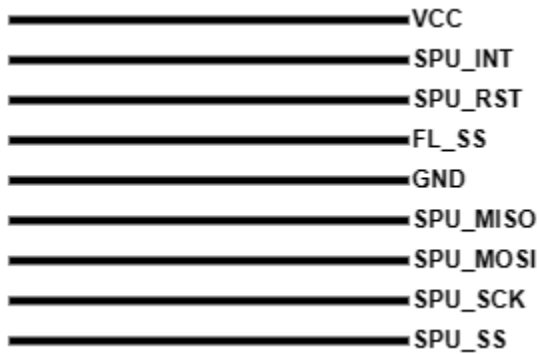


If using the jumper wire kit, use the following diagrams to connect the small EVB board to the Tympan host.





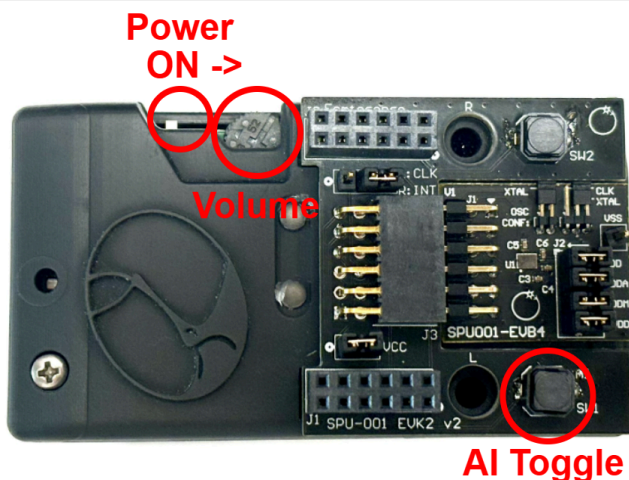
**Wiring diagram of the SPU-001 EVB (side view looking into the 12-pin header). Connect the circled SPU-001 EVB pins to their corresponding host pins.**



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

## 1.2. Run the built in example

Once the steps above are completed, flip the power switch to the RIGHT position to turn it on. The LED will quickly blink red or green 1 time when powered up. If the EVK 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 10 minutes.



**The power switch and AI toggle switch are circled in red. For demos with audio output (e.g. AINR), the volume knob controls the volume level. For the jumper wire kit, use the wired button to toggle AI processing on and off**



The program will take a few seconds to boot. After that, you should then see the green LED on top of the host quickly flash 3x. The demo will then start according to the installed firmware below.

### 1.2.1. AI-Noise Reduction (AINR) demo

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



You can toggle the AINR processing on and off using the button marked SW1 or AI 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.

### 1.2.2. Speech Commands Demo

The speech commands demo responds to speech commands. The demo will be set to either “Alexa” or “Google Speech Commands” (GSC). When you say the command, the LEDs will flash the number of times corresponding to the command that is spoken. The command will also be displayed in the serial terminal (115200 baud 8N1).

For the Alexa demo, the light will flash once when you say “Alexa”

For the GSC demo, the following commands are used:

- 1: "YES"
- 2: "NO"
- 3: "ON"
- 4: "OFF"

- 5: "LEFT"
- 6: "RIGHT"
- 7: "UP"
- 8: "DOWN"
- 9: "STOP"
- 10: "GO"
- 11: "HEY SNIPS"

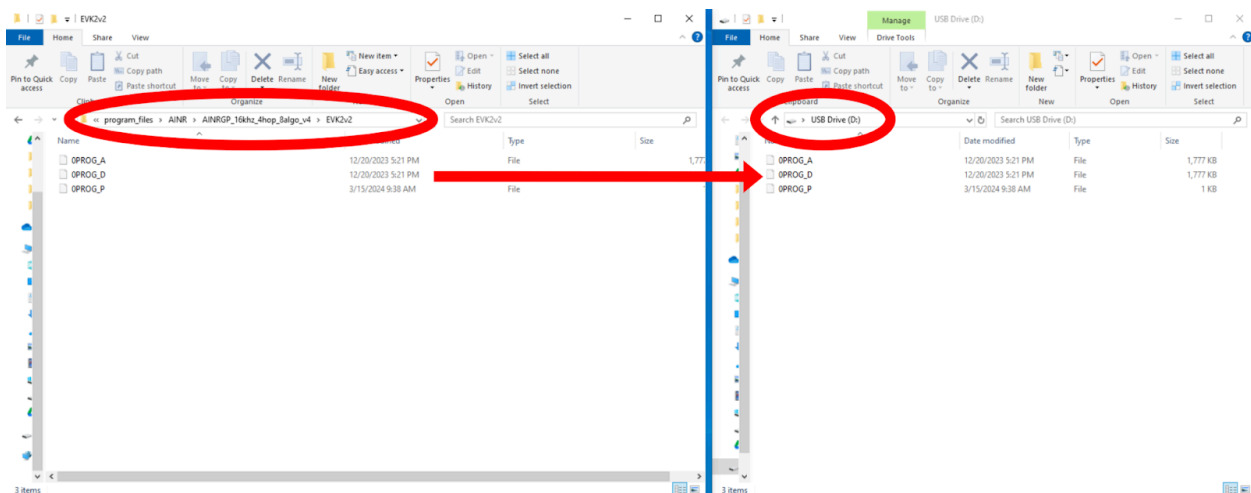
Pressing the AI toggle button turns off the AI processing, and the red LED will illuminate. Press it again to enable AI processing.

### 1.3. Changing the model and/or firmware

In order to change the firmware to use a different Femtosense model, first navigate to the directory of the model you want to work with. For example, the 8ms latency AINR model is located in:

`program_files/AINR/AINRGP_16khz_4hop_8algo_v4/`

Select the folder within that matches your EVK version, EVK2v1 or EVK2v2. Copy the 3 files `0PROG_A`, `0PROG_D`, and `0PROG_P` to the SD card located in the EVK as shown below.

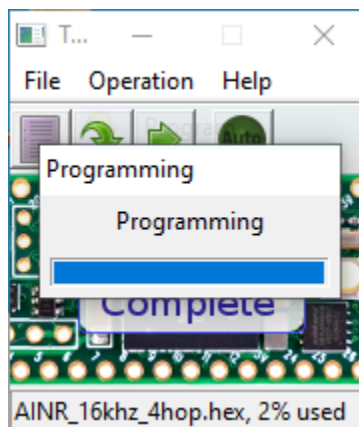




After loading the files, reinsert the SD card with the metal contacts facing up. Next, download the Teensy 4.1 firmware loader tool for your OS from: <https://www.pjrc.com/teensy/loader.html>. Open the program, and select **File > Open HEX File** and select the .hex file located in the model directory, for example:

`program_files/AINR/AINRGP_16khz_4hop_8algo_v4/AINR_16khz_4hop.hex`

Connect the EVK to your computer with the USB cable, turn it on, then press the PROGRAM FIRMWARE button shown above on the EVK. Complete the programming by selecting **Operation > Program**, then **Operation > Reboot**. The EVK should reboot into the new firmware.



This works the same way for the speech commands models.

## 1.4. Storing Multiple Models on the SD card

Multiple types of models that use the same firmware can be loaded onto the sd card by adjusting the number in the file name. For example, 4 models on the SD card would contain the following 12 files:

- 0PROG\_A
- 0PROG\_D
- 0PROG\_P
- 1PROG\_A
- 1PROG\_D
- 1PROG\_P
- 2PROG\_A
- 2PROG\_D
- 2PROG\_P
- 3PROG\_A
- 3PROG\_D
- 3PROG\_P

The **XPROG\_P** file is optional, and the defaults in the next section will be used if this file does not exist. In order to switch between the models, do the following:

1. Hold down the SW1 button.
2. Switch on the EVK
3. Both LEDs will illuminate, indicating that model selection mode is active. Release the button.
4. The LEDs will flash once, indicating that **1PROG\_X** is selected.
5. To confirm model 1, hold down SW1. 1 blink will confirm that model 1 is selected
6. To select model 2, tap SW1 briefly. Each time you tap SW1, the next model will be selected, indicated by the number of blinks. Hold SW1 to select the model.

If you do not hold down the SW1 button while turning on the EVK, the default **0PROG\_X** files will be used.

## 1.5. XPROG\_P Parameters

Many parameters can be used to adjust the firmware, which can be loaded into the **XPROG\_P** file on the SD card. The format is shown in the following screenshot:

```

1 SPI_SPEED 16000000
2 VCO_MHZ 44
3 USE_MEMORY_FSM 1
4 CLKOD 4
5 DISABLE_CORE1 0
6 INPUT_GAIN_DB 30.0
7 OUTPUT_MAX_GAIN_DB 0.0
8 USE_EXTERNAL_MIC 0
9 USE_ONBOARD_XTAL 1
10 USE_INT_PIN 1
11 CONFIRM_MODEL_TYPE SC
12 CONFIRM_2MIC 0
13 CONFIRM_HOP_SIZE 128
14 CONFIRM_ARCHITECTURE MP
15 COMMAND_NAME 1 YES
16 COMMAND_NAME 2 NO
17 COMMAND_NAME 3 ON
18 COMMAND_NAME 4 OFF
19 COMMAND_NAME 5 LEFT
20 COMMAND_NAME 6 RIGHT
21 COMMAND_NAME 7 UP
22 COMMAND_NAME 8 DOWN
23 COMMAND_NAME 9 STOP
24 COMMAND_NAME 10 GO
25 COMMAND_NAME 11 Hey Snips!

```

*Example of the XPROG\_P format for a speech commands model with 11 commands.*

These parameters are described in the table below. If you generate your own models, you should also construct a similar **XPROG\_P** file to set up the firmware to receive your model correctly. More information about optimizing these parameters can be found in the SPU Integration Guide document.

| Parameter                   | Description  | EVK2v1 ONLY | AINR Support | SPEECH COMMANDS Support | Min              | AINR Default | SPEECH COMMANDS Default | Max |
|-----------------------------|--|-------------|--------------|-------------------------|------------------|--------------|-------------------------|-----|
| <b>CLKOD</b>                | Output Clock Divider with respect to VCO_MHZ. Must be 1 or even.   | No          | Yes          | Yes                     | 1                | 1            | 1                       | 16  |
| <b>COMMAND_NAME X</b>       | Individual command name  | No          | No           | Yes                     | See Next Section |              |                         |     |
| <b>CONFIRM_2MIC</b>         | Sets the firmware to send two mic inputs to the model (0=1mic, 1=2mic)   | No          | Yes          | Yes                     | 0                | 0            | 0                       | 1   |
| <b>CONFIRM_ARCHITECTURE</b> | Checks if the architecture matches the EVK hardware.. Firmware will error if this differs from the firmware value. | No          | Yes          | Yes                     | {TC2, MP}        | MP           | MP                      | -   |
| <b>CONFIRM_HOP_SIZE</b>     | Checks if the hop size is the expected value. Firmware will error if this differs from the firmware value          | No          | Yes          | Yes                     | 0                | in ANR.h     | In or Speech Commands.h | 256 |
| <b>CONFIRM_MODEL_TYPE</b>   | Checks if the model type is the expected value. Firmware will error if this differs from the firmware value.       | No          | Yes          | Yes                     | {AINR, SC}       | AINR         | SC                      | -   |



|                      |  |     |     |     |          |           |          |            |
|----------------------|--|-----|-----|-----|----------|-----------|----------|------------|
| CORE_GATING_EN       | Enable Core Clock Gating (0=off, 1=on)   | Yes | Yes | Yes | 0        | 0         | 0        | 1          |
| CORE_TRANSFER_US     | Core Clock Gating Parameter (us)   | Yes | Yes | Yes | 0        | 0         | 0        | -          |
| CORE0_READY_US       | Core Clock Gating Parameter (us)   | Yes | Yes | Yes | 0        | 0         | 0        | -          |
| DISABLE_CORE1        | Turn off second core if not needed (0=core1 enabled, 1=core1 disabled)                                       | No  | Yes | Yes | 0        | 0         | 0        | 1          |
| ENABLE_GUI           | Send Femtosense GUI control commands through serial terminal (0=dont send GUI commands, 1=send GUI commands) | No  | No  | Yes | 0        | -         | 0        | 1          |
| INPUT_GAIN_DB        | Input gain from microphone input to SPU (dB, 0.5d B steps)   | No  | Yes | Yes | 0        | 30.0      | 30.0     | 47.5       |
| OUTPUT_MAX_GAIN_DB   | Maximum output gain on audio output jack (dB)  | No  | Yes | No  | -20.0    | 0         | -        | 40.0       |
| PRINT_EXECUTION_TIME | Turn on Serial printing of the model execution time in microseconds (0=off, 1=on)                            | No  | Yes | Yes | 0        | 0         | 0        | 1          |
| SAFETY_MARGIN_US     | Core Clock Gating Parameter (us)   | Yes | Yes | Yes | 0        | 0         | 0        | -          |
| SPI_SPEED            | SPI Speed between SPU and Tympan (Hz)  | No  | Yes | Yes | 4000 000 | 16000 000 | 4000 000 | 5000 0 000 |
| TOTAL_WAIT_TIME_US   | Total wait time before checking for SPU interrupt (us)   | Yes | Yes | Yes | 0        | 0         | 0        | -          |
| USE_EXTERNAL_MIC     | Use external mic (0=internal mic, 1=external mic jack, 2=mic header)   | No  | Yes | Yes | 0        | 0         | 0        | 2          |
| USE_INT_PIN          | Use asynchronous hardware interrupt pin (0=use register interrupt, 1=use pin interrupt)                      | No  | Yes | Yes | 0        | 1         | 1        | 1          |
| USE_MEMORY_FSM       | Enable memory power-saving mode on SPU (0 or 1)  | No  | Yes | Yes | 0        | 0         | 0        | 1          |
| USE_ONBOARD_XTAL     | Use onboard crystal instead of external reference clock (0=external clock, 1=crystal)                        | No  | Yes | Yes | 0        | 1         | 1        | 1          |
| VCO_MHZ              | SPU Clock Speed (Mhz)  | No  | Yes | Yes | 30       | 250       | 100      | 268        |

Note that the parameters distributed with Femtosense models should not be adjusted, as they are already optimized for performance and low power consumption.

The **COMMAND\_NAME X** parameter is used to set the command names for speech command algorithms. For each command, add a line to **XPROG\_P** in the following format:

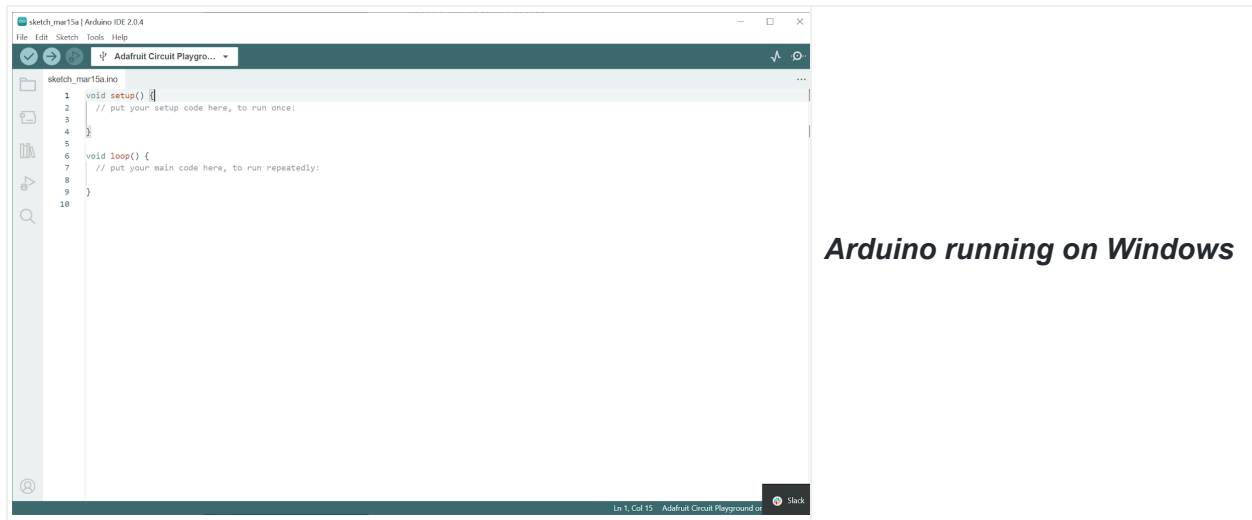
**COMMAND\_NAME** <command index, starting with 1> <command name>

An example of this format is given in the screenshot earlier in this section.

## 1.6. Setting up the Firmware Development Environment

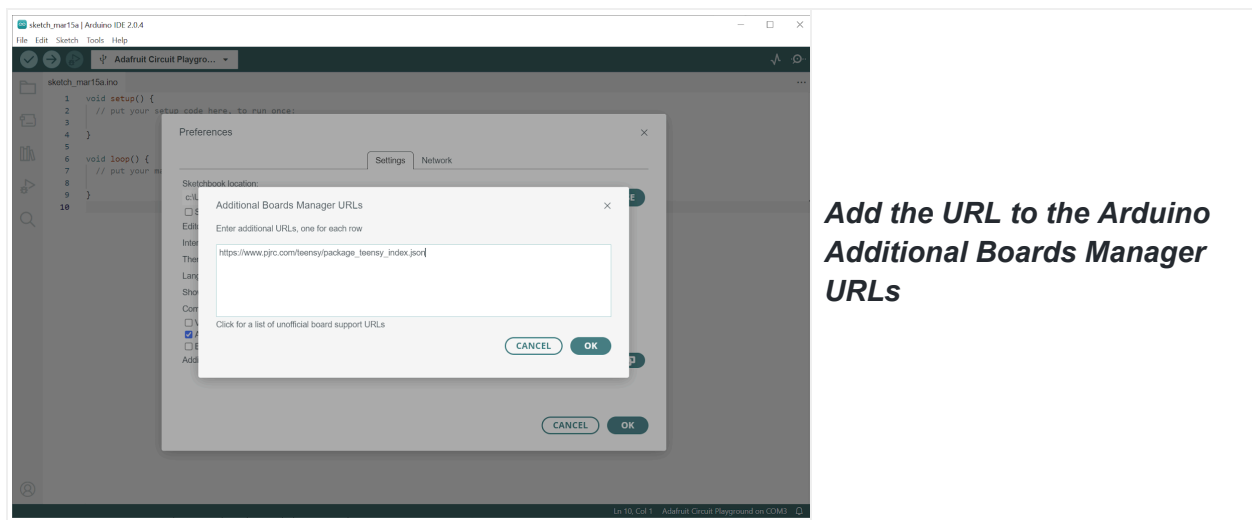
The EVK2 firmware is open source and can be adjusted as needed using the Arduino platform. The following instructions are for Windows, but a similar process can be followed for Mac OS or Linux operating systems. However, we have not tested Linux support.

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

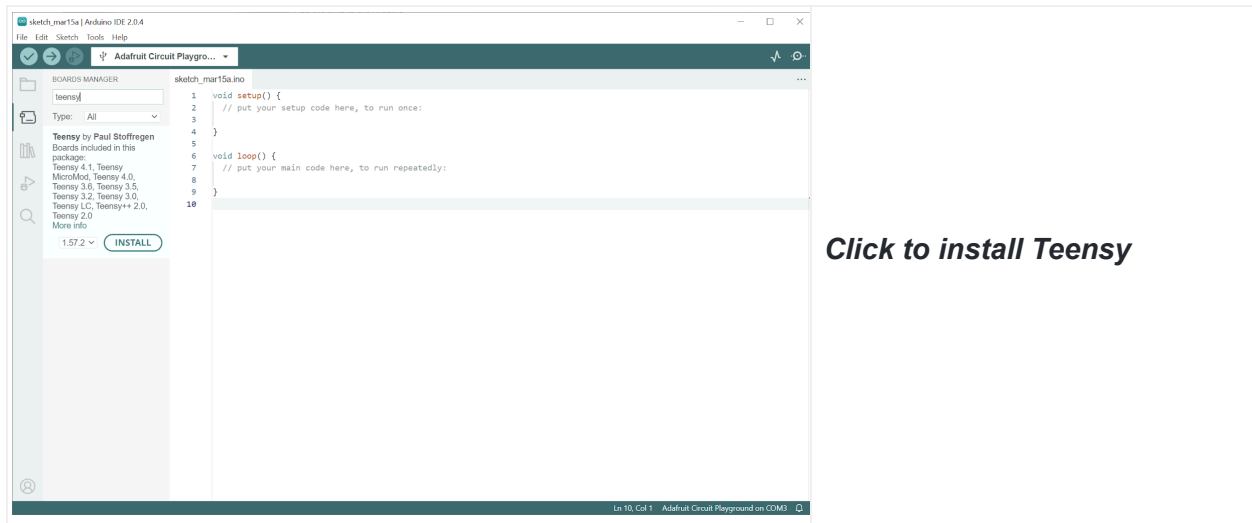


Next, go to **File > Preferences** in Arduino, and click the button next to “Additional board manager URLs”. Add the following line to the URLs:

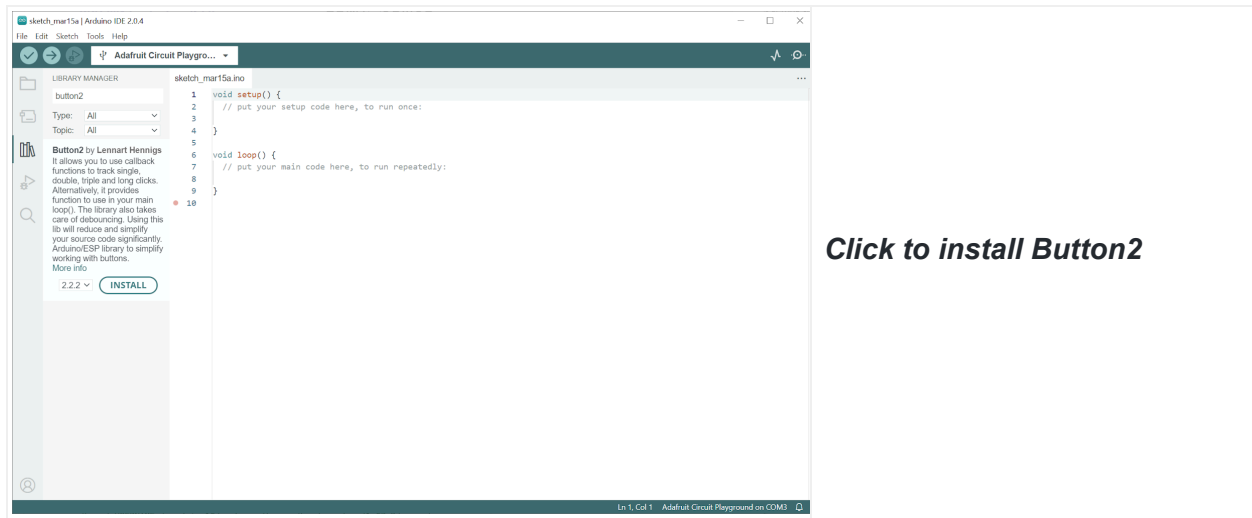
[https://www.pjrc.com/teensy/package\\_teensy\\_index.json](https://www.pjrc.com/teensy/package_teensy_index.json)



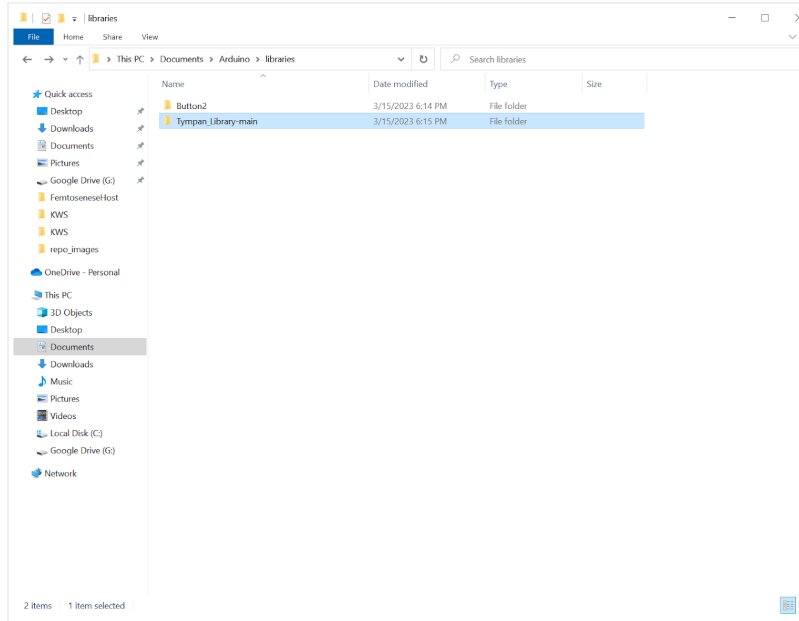
Click OK to go back to the Arduino IDE. Next, click the “Boards Manager” icon on the left side of the IDE and search for “Teensy”. Install the package “Teensy by Paul Stoffregen”. Choose version 1.57.2 from the dropdown as shown below.



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



Next, download the Tympan library from Github at: [https://github.com/Tympan/Tympan\\_Library](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.***

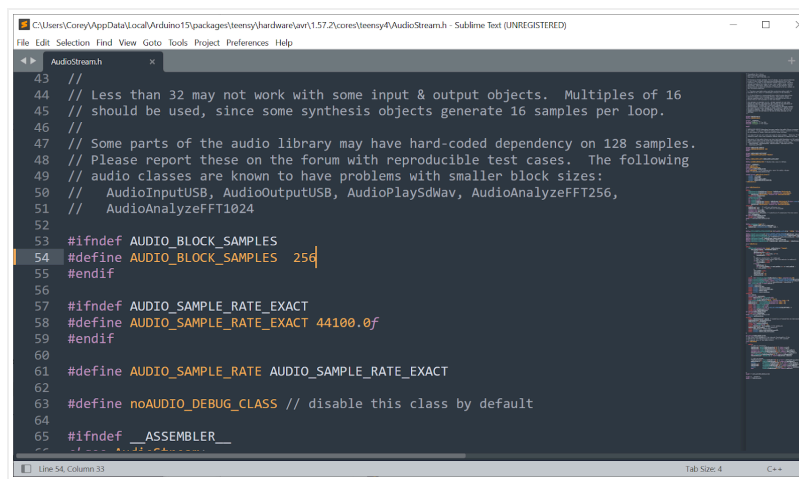
Next, for some firmwares, you will need to modify one library file, which in Windows is located at:

`C:\Users\<USERNAME>\AppData\Local\Arduino15\packages\teensy\hardware\avr\1.57.2\cores\teensy4\AudioStream.h.`

Change:

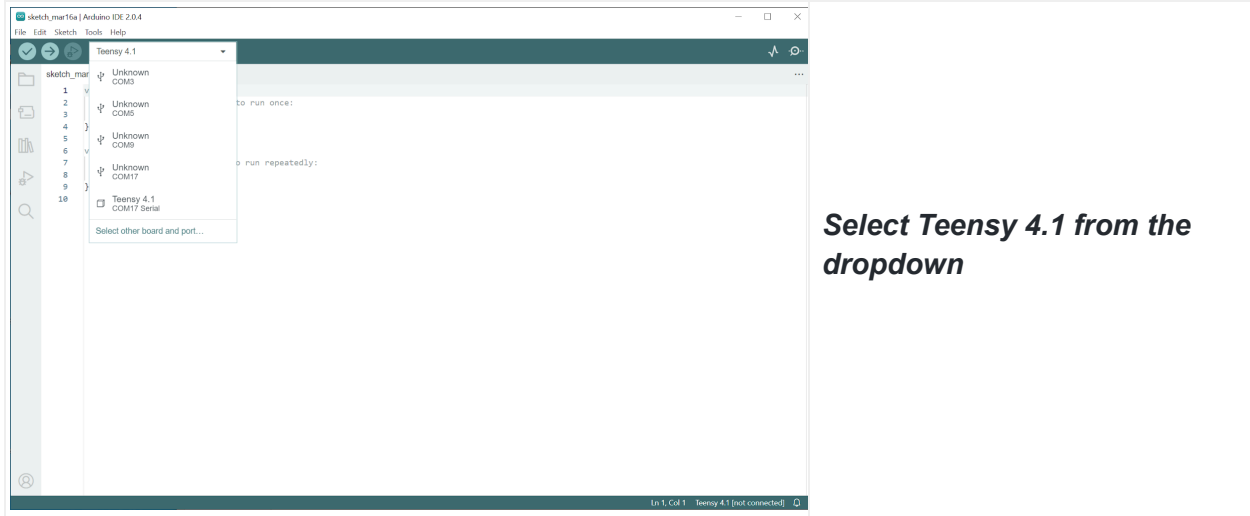
`#define AUDIO_BLOCK_SAMPLES`

from `128` to `256` and save the file. Note that you will need to redo this change if the Teensy library is updated.



***Update the value for AUDIO\_BLOCK\_SAMPLES to 256***

Next, close and reopen Arduino. This will make sure that all of the changes above are loaded. Now, 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.



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

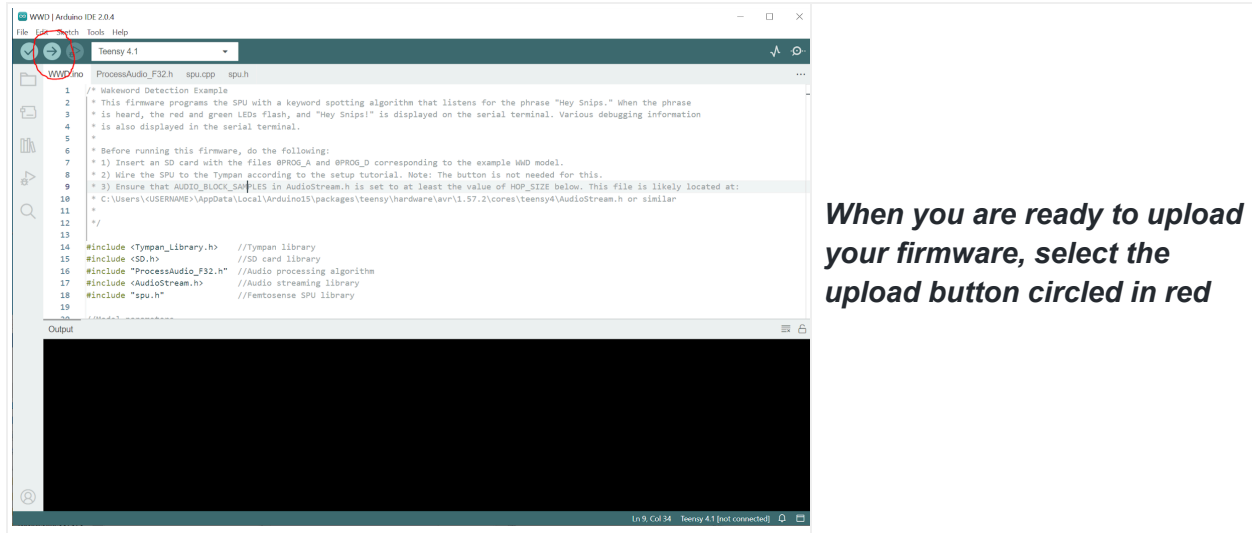
## 1.7. Uploading the Firmware from Arduino

To upload firmware, first open the target firmware's **.ino** file in Arduino. For example, the AINR firmware is located in:

`program_files/AINR/src/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 which includes the SPU API and audio processing functions. Note that in order to change the sample rate or hop size, you need to adjust these parameters in the min header file, either **AINR.h** or **SpeechCommands.h**. 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.



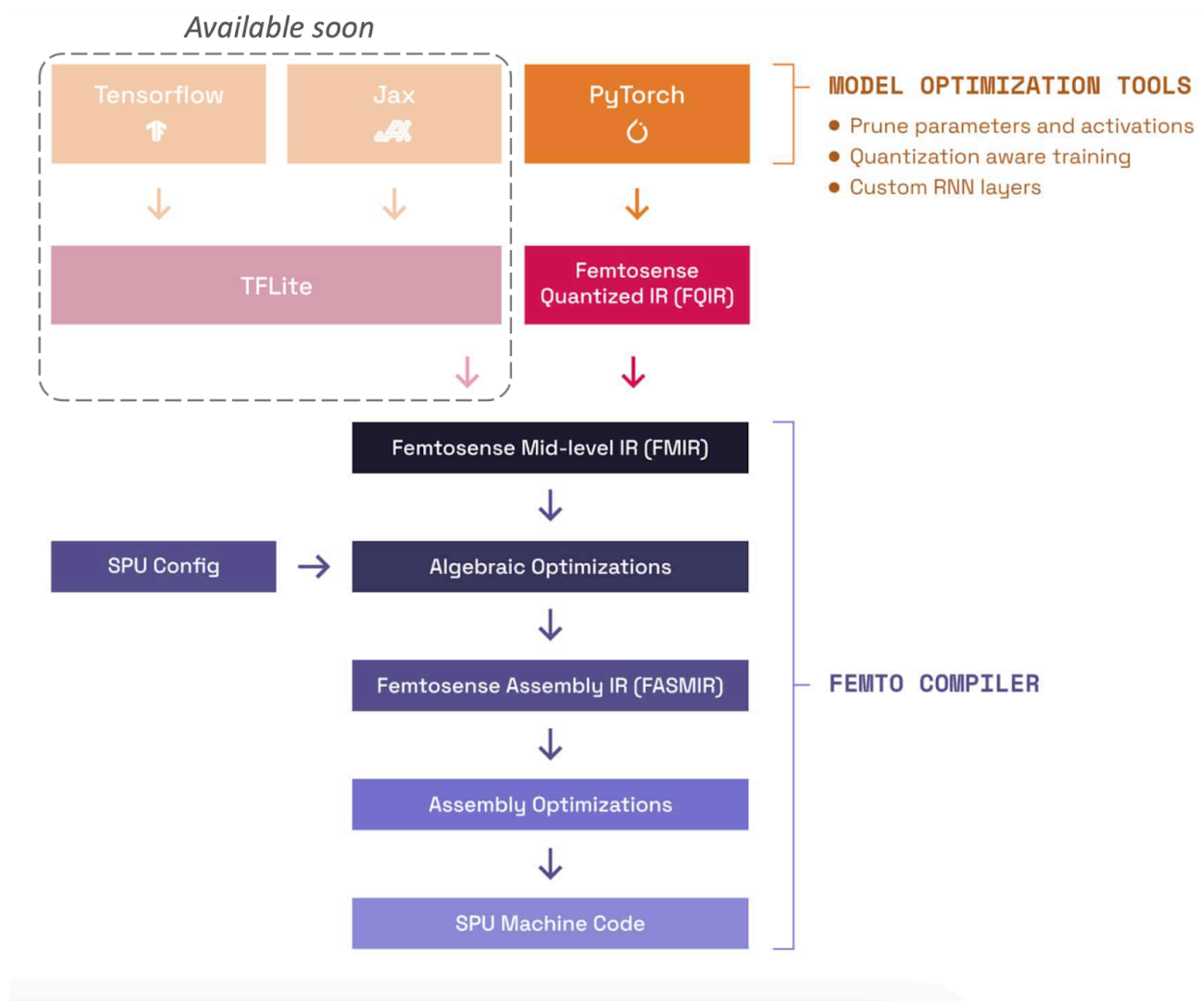


## 2. For Machine Learning Developers

### 2.1 The SPU Development Kit

FemtoseNSE 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 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.**

**Disclaimer:** we recommend using the PyTorch (fmot) frontend for model porting, as it's internally developed, robust, widely tested, and compatible with our femtocrux compiler.

For TFLite frontend users, only femtocrux version 0.3.0 is compatible. Note that TFLite frontend updates and patches will be suspended until further notice due to compatibility and flexibility issues. We strongly recommend converting your TFLite models to PyTorch to prevent future compiler tooling issues. Contact [techsupport@femtosome.ai](mailto:techsupport@femtosome.ai) for assistance or feature inquiries.

## 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 <sup>1</sup>  |
|--------------------------------|--------|--|---|
| <a href="#">fmot</a>           | PyPI   | The PyTorch frontend for FemtoseNSE                  | <a href="https://fmot.femtoseNSE.ai/">https://fmot.femtoseNSE.ai/</a>           |
| <a href="#">femtoCrux</a>      | PyPI   | The FemtoseNSE compiler                              | <a href="https://femtoCrux.femtoseNSE.ai/">https://femtoCrux.femtoseNSE.ai/</a> |
| <a href="#">femtoDriverPub</a> | 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 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](#).

---

<sup>1</sup> 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  |
|---|--|
| <b>green 3x</b>                             | Bootup finished normally   |
| <b>Red or green 1x</b>                      | A quick flash at bootup indicates if the <code>XPROG_P</code> parameters file was found (green) or not (red). Default parameters will be used if not found   |
| <b>Red + green</b>                          | If both LEDs are on, this means you are in model selection mode. Hold SW1 to select the current file.  |
| <b>red 2x</b>                               | 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> (16KHz audio) or <code>HOP_SIZE * 2</code> (8KHz audio) This file is located at:<br><code>C:\Users\&lt;USERNAME&gt;\AppData\Local\Arduino15\packages\teensy\hardware\avr\1.57.2\cores\teensy4\AudioStream.h</code> or similar in the Windows operating system. |
| <b>red 3x</b>                               | This indicates that the SPU integrity or clock check has failed. Check the connection between the SPU-001 circuit board and the Tympan host. Also, check that all jumpers are correctly installed.   |
| <b>red 4x</b>                               | This indicates that the programming files cannot be read from the SD card, or the data in <code>XPROG_P</code> is invalid. Make sure that the SD card is inserted and that the two required programming files (e.g., <code>XPROG_A</code> and <code>XPROG_D</code> ) are in the root directory, and that <code>XPROG_P</code> values are not invalid.  |
| <b>red 6x</b>                               | This indicates that the SPU has malfunctioned during processing. Check the <code>XPROG_P</code> parameters. If this error occurs immediately after bootup, check that the model files on the SD card are correct.  |
| <b>red 7x</b>                               | This indicates that the model values in <code>XPROG_P</code> do not match the firmware. Use the correct model  |
| <b>Constant green or button not working</b> | 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.  |
| <b>No LED or red LED</b>                    | This means the model is off and static power can be measured.  |

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 Examples

#### 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 `XPROG_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 `XPROG_P` in order to set the max headphone volume level. The maximum value for this parameter is 40.

### 3.3. Speech Command Examples

#### Objective

The green and red LEDs on the Tympan should blink the number of times corresponding to the command index in the model. For the Alexa models, the command “Alexa” is command 1. For the GSC models, section 1.2.2 of this document shows the command indices. The command is also shown in the serial terminal (115200 baud 8N1)

#### Caveat

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

#### Troubleshooting

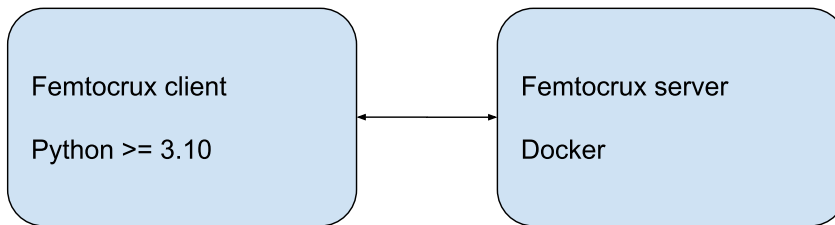
The provided firmware sets the microphone gain level to a good initial value. Still, you can adjust the firmware to change microphone gain if desired by adjusting the `INPUT_GAIN_DB` parameter in `XPROG_P`.



## 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

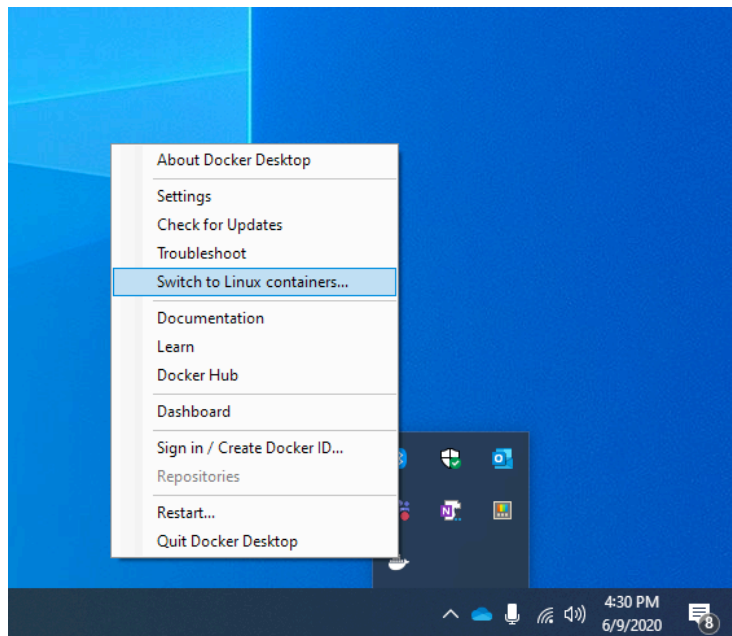
First, install Docker Desktop by following these [instructions](#).

- 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  $\geq 3.10$ ) 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/femtosome/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/femtosome/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<br>Added section 1.7 - guide to revert to WWD firmware<br>Added section 1.8 - 0PROG_P parameter table<br>Added Appendix A.1 - Femtocrux Windows Setup Guide |
| 1.7     | 2023-10-18   | Fixed missing info in wiring diagram, added 256 hop fix back, added GSC info, updated 0PROG_P parameters table, updated error code table, removed WWD video (blink pattern has changed)                           |
| 1.8     | 2023-12-28   | Added Alexa support   |
| 2.0     | 2024-03-18   | Added EVK2v2 support, multiple model selection support, and binary firmware update instructions   |