Development of an Open Source Audio Processing Platform

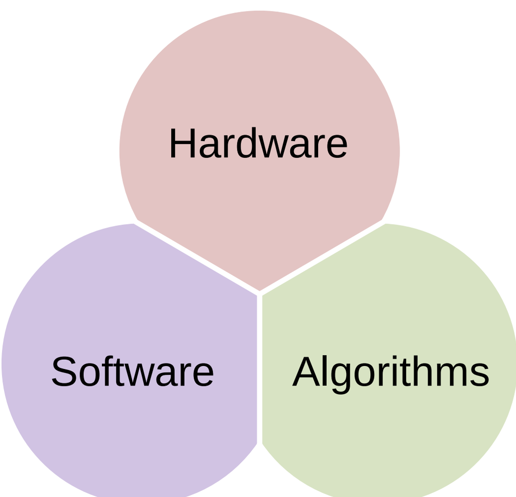
William (Chip) Audette, Odile Clavier, Creare LLC

Daniel Rasetshwane, Stephen Neely, Boys Town National Research Hospital

Joshua Alexander, Purdue University

MOTIVATION

Real-time audio processing is an important tool for acoustic and hearing research. But, like many of us, you never test your ideas in real time. Why?



Real-time is hard. Hardware, software, and algorithms—you need skills in all three. It is too big a hurdle.

We seek to lower this hurdle so that real-time processing is part of everyone's toolbox.

APPROACH

Our approach is to develop a complete system—hardware, software, and algorithms—that can get you started. Having a working system lowers the initial hurdle that has kept you from using real-time processing in your research.

Open Source: Certainly, our system does not do exactly what you want it to do. So, we shared our system as open source. You can see what we have done, learn from it, and adapt it to your needs. You do not need to start from scratch. Sharing speeds innovation for everyone.

HARDWARE

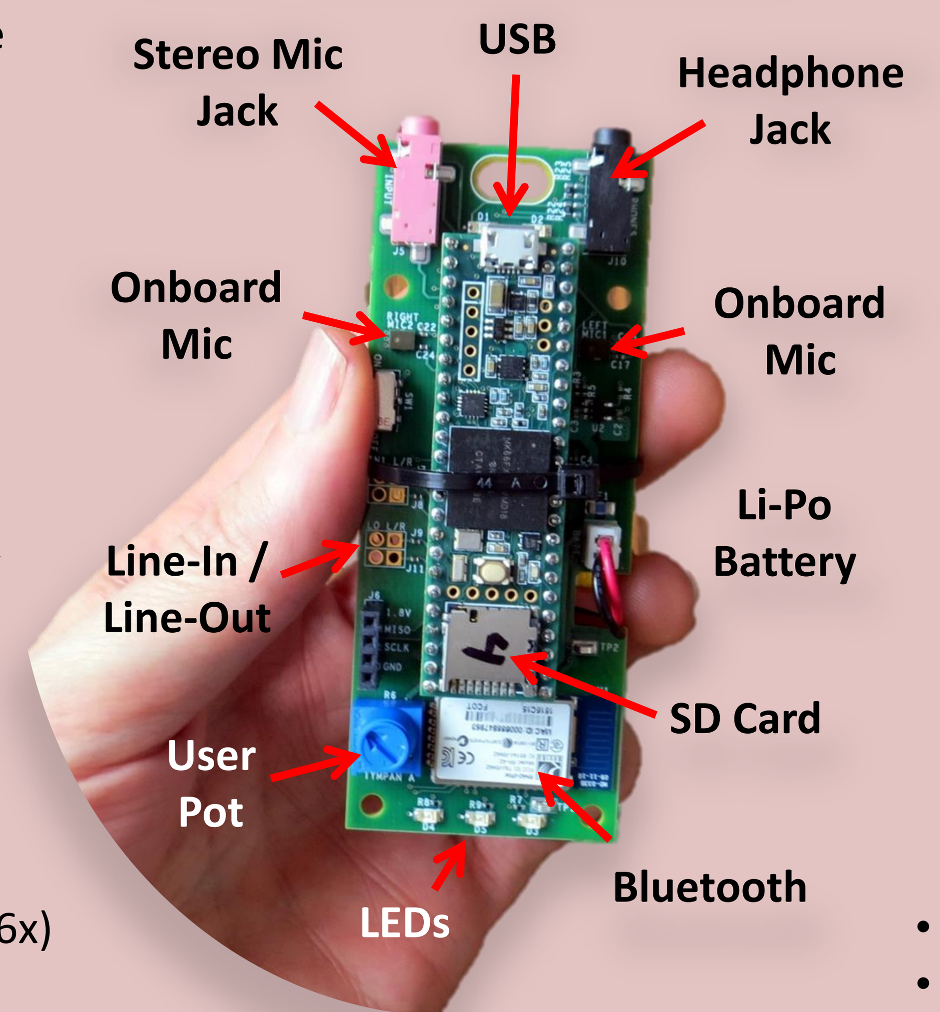
Real-time audio processing requires real-time hardware. The hardware must acquire audio signals cleanly. It must be easily programmed. It must be affordable.

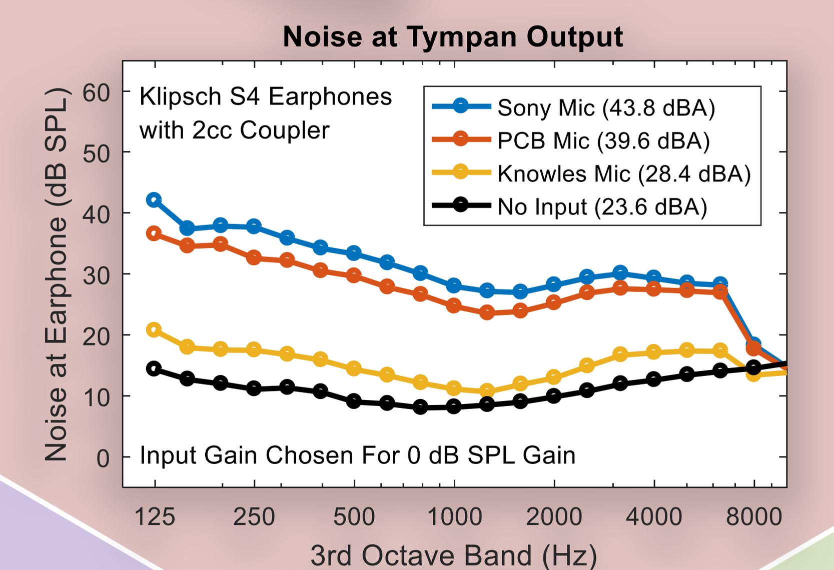
Our approach is to start from an existing, low-cost, digital processing board. We then extend the processing board by adding our own custom audio interface to handle your audio signals.

Processor: The processing board is a Teensy 3.6, which is small, inexpensive, and includes features like USB and an SD card. It is programmable via the Arduino IDE, getting started is much easier.

Processor Features

- ARM M4F Core (NXP K66x)
- 180 MHz, 32-bit
- Floating-point unit, DSP
- USB Audio / MIDI / Serial
- SD card for audio or data





Audio Interface: To get audio into and out of the Teensy, we designed a daughter board to host an audio codec (TI TLV320AIC3206). The codec provides the ADC/DAC functions, gain, mic bias, and headphone amplifiers.

In addition to the codec, our custom board includes onboard microphones, connectors for external mics and earphones, battery charging, LEDs, Bluetooth, and a programmable knob.

Open Source: The hardware design is open source. So, if you don't want to buy it from us, you can make one yourself. Or, even better, you can modify our design to better suit your own needs. Then share it with the community!

Audio Interface Features

- Stereo inputs and outputs
- Programmable gain
- Headphone amplifier
- Sample rate up to 192 kHz
- Dynamic range of 95 dBA.

NEXT STEPS

In this multi-year effort, a new version of the hardware and firmware will be released yearly. Next steps include:

- Faster: We feel constrained by the current processor. Upgrading would enable more complex algorithms.
- Algorithms: More example algorithms help all users. Look for adaptive filtering and feedback cancellation.
- Earpieces: To improve our hearing aid studies, we will develop custom earpieces for mics and speakers.
- Smaller: We would like to shrink the device so that it could be worn on the head, instead of on the body.

PARTICIPATE

Our hardware is called "Tympan" and it is about to be released. Sign up at tympan.org. Join the forum to ask questions or to share project ideas. What would you like to do with the system?

Our designs, software, and algorithms are available at github.com/tympan. Do you have your own algorithm ideas? Fork our repo and contribute! Your ideas will help to make it better.

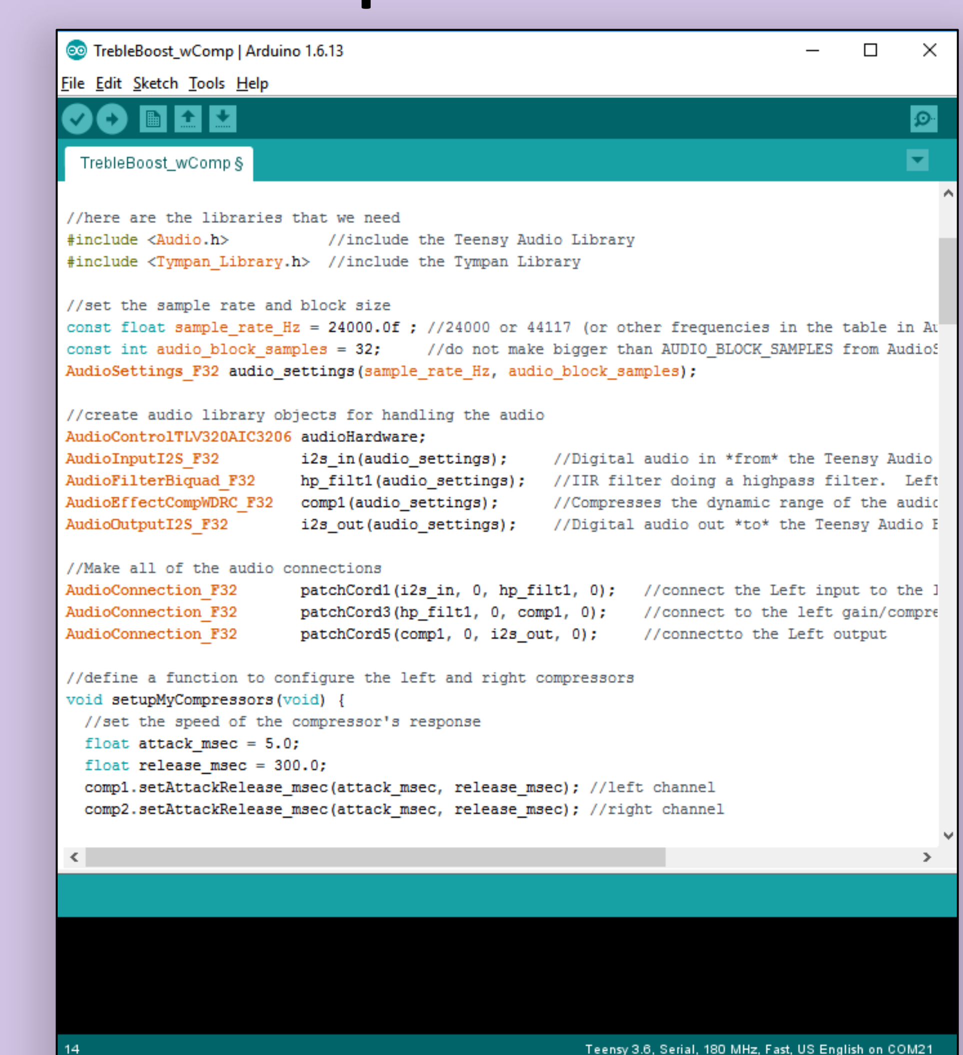
tympan.org

SOFTWARE

Traditionally, real-time digital audio systems could only be programmed by professionals. It required specialized expertise and expensive tools. It was a major hurdle that often prevented its use by acoustics and hearing researchers.

Through the success of open source communities such as Arduino, this has all changed. We now know that novices, even hobbyists, can program digital hardware. How did they do it? Be open, provide good examples, encourage questions, share experience.

Developer Environment



Arduino IDE: Our hardware is programmed through the popular Arduino development environment ("Arduino IDE"). By following their tutorials and examples, you quickly see how to use the Arduino IDE to program digital devices like ours. The language is C/C++. The Arduino IDE is free.

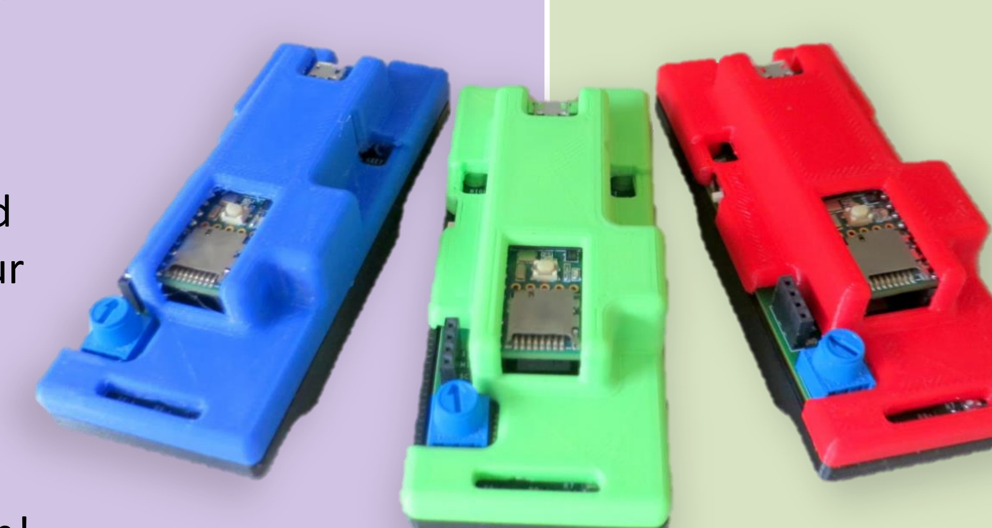
Tympan Library: Like most Arduino add-ons, access to our audio hardware is encapsulated in a free, open, Arduino library. Start with our examples and you'll see how quickly you can make sound flow through the system.

After a little success, you may even find that programming real-time audio hardware is fun!

ARDUINO

GitHub

3-D Printed Enclosures



ALGORITHMS

We expect that most people will use our system as a platform for trying their own audio processing algorithms. In addition to good hardware and software, you probably also need help with getting your algorithm to work in real time. Real-time processing is a challenge. We needed to lower that hurdle, too.

Building Blocks: In our open-source library, we provide a number of basic processing functions such as gain, IIR/FIR filters, level detectors, mixers, and FFT/IFFT operators.

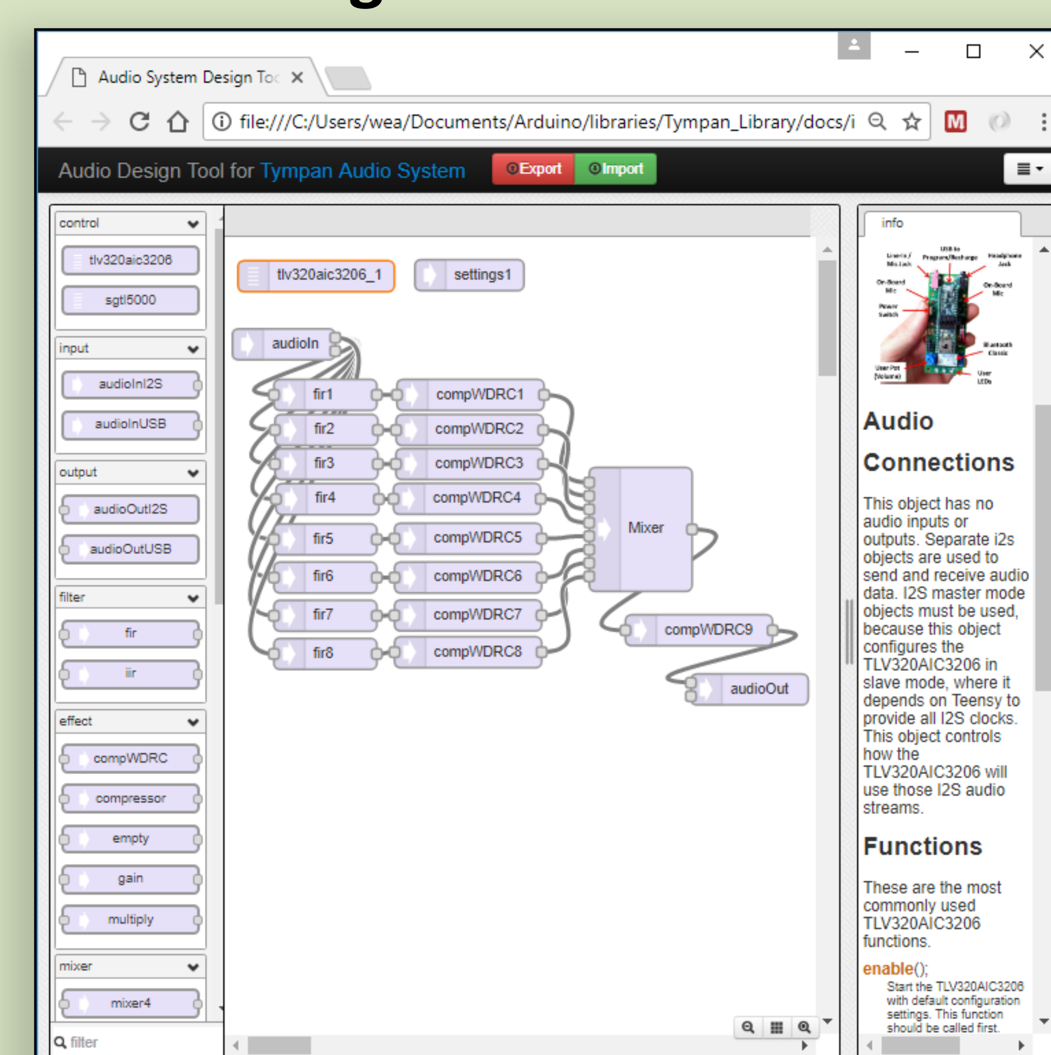
Following our examples, you can mix together your own algorithm from these basic ingredients. Or, you can build your algorithm from scratch. Or, do both.

Algorithm Starter: We know that starting a new algorithm is hard. A blank page is demotivating. How do I start?

We created an "Algorithm Starter" tool that lets you graphically build your algorithm from parts. Click "export" and it writes an outline of your code that you paste into the Arduino IDE and finish yourself.

Like you, we come from MATLAB/Python. We all need help when moving to real time.

Algorithm Starter



This work has been supported by the National Institutes on Deafness and Communication Disorders under SBIR Phase I Grant #1R44DC015445-01