

Open-source maker platform
for beautifully responsive
interactive audio



Adan L. Benito

@BelaPlatform

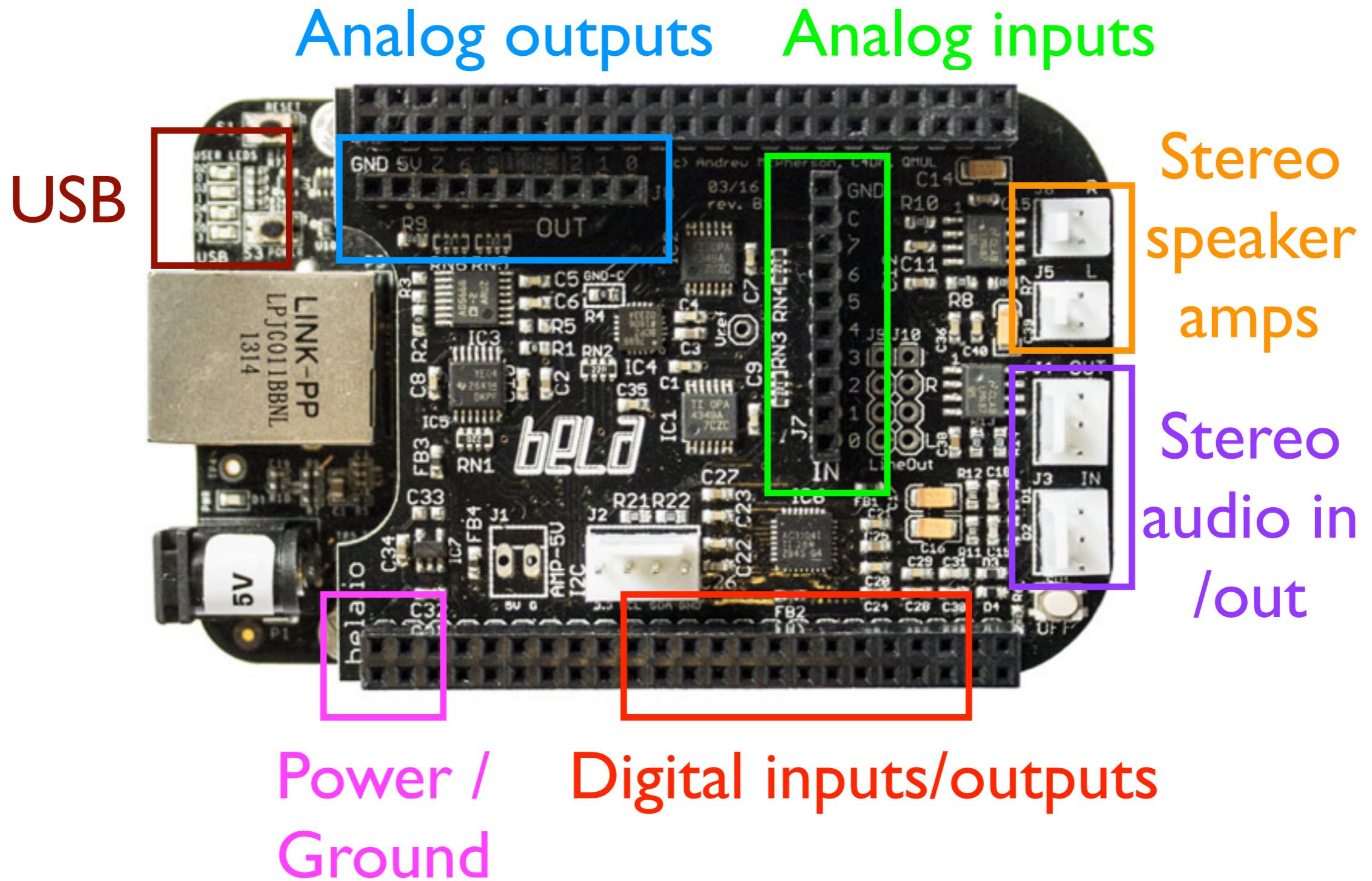


Programmable Audio Workshop

1st of December, 2018

GRAME-CNCM, Lyon

Bela hardware



Required software

Firefox/Chrome browser

The IDE may or may not work in other browsers.



Getting started

1. Plug in Bela to computer (USB)
2. Launch the browser
3. After around 30s go to the url:

MacOS, Linux

<http://bela.local/>

Windows

<http://192.168.6.2>

Troubleshooting: drivers

On Mac Yosemite and above, Windows Vista and above, and all Linux no drivers are required.

For older operating systems the drivers can be found on the board

Troubleshooting: drivers

Step 1: Plug in Bela by USB

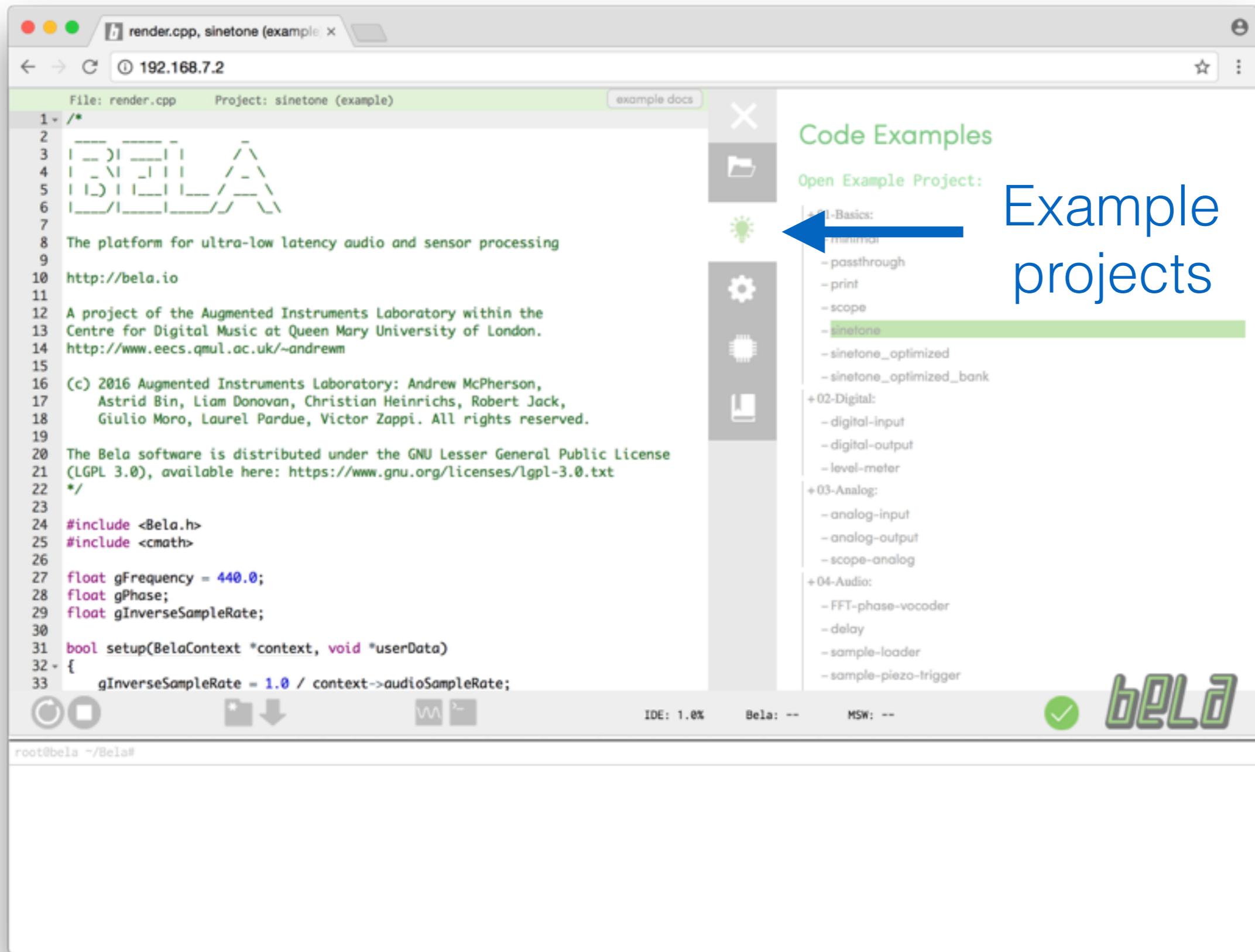
Wait for it to boot

You should see a drive **BELABOOT** come up

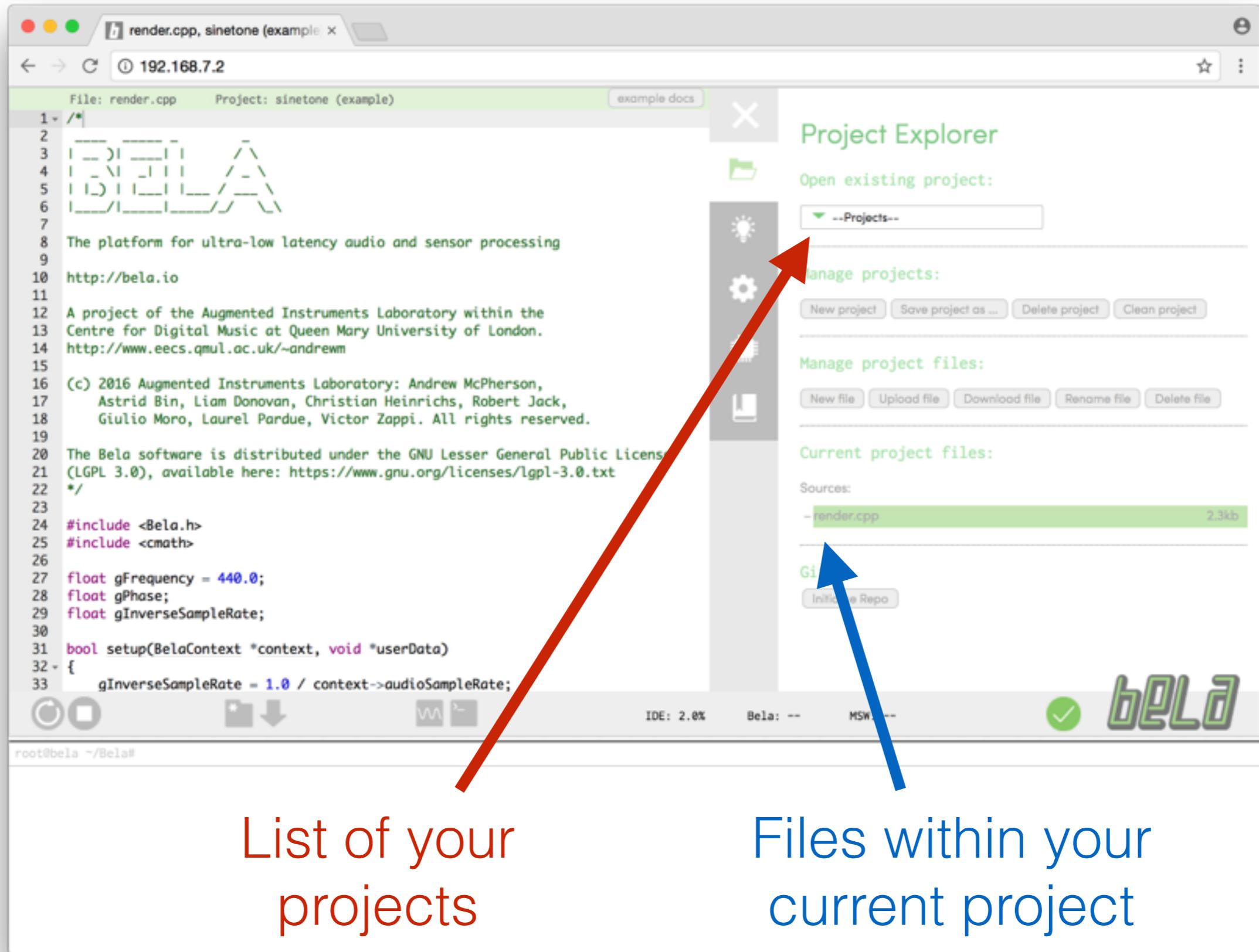
Step 2: Install software from Drivers directory

Step 3: Reboot computer

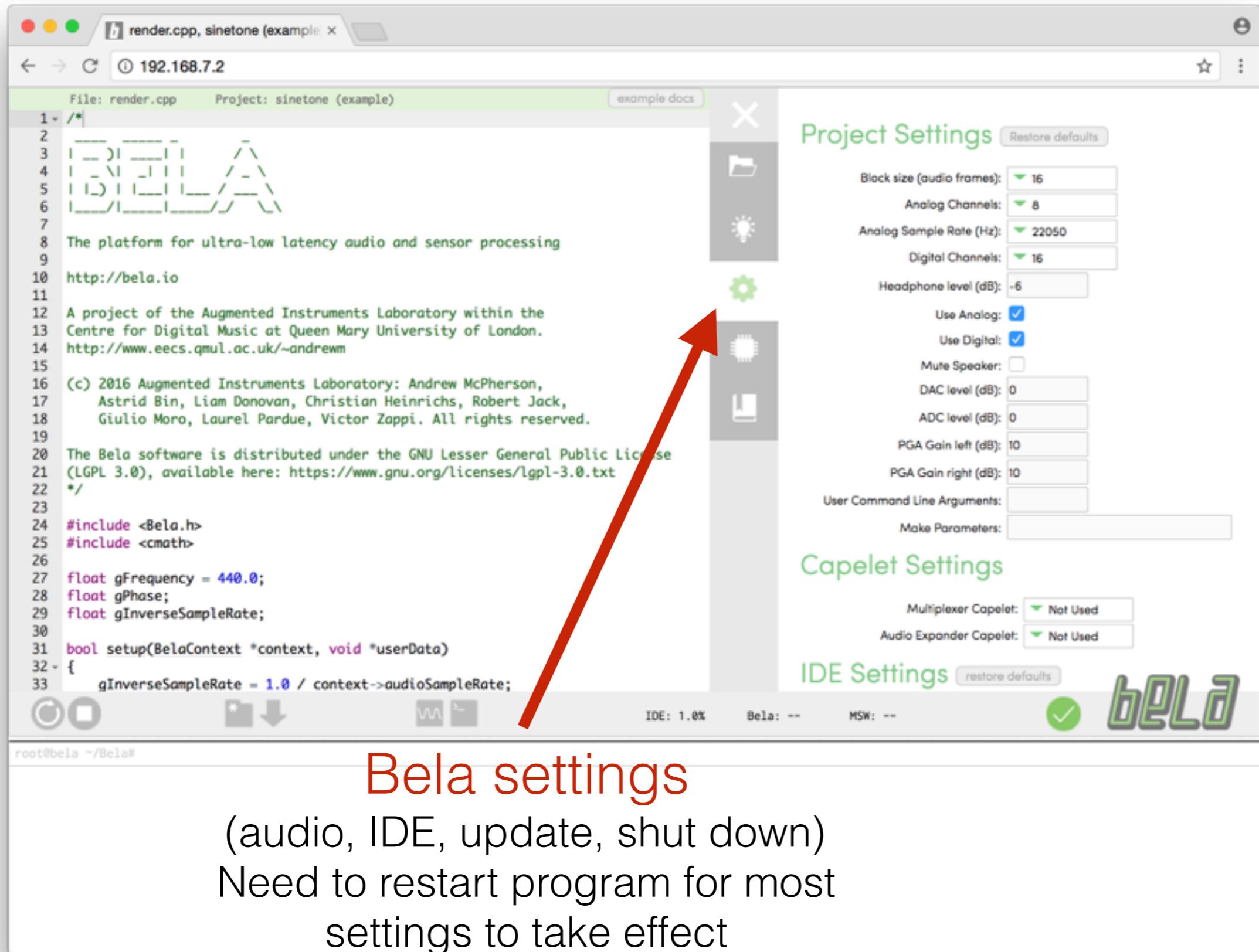
Bring up the Bela IDE: <http://bela.local/>



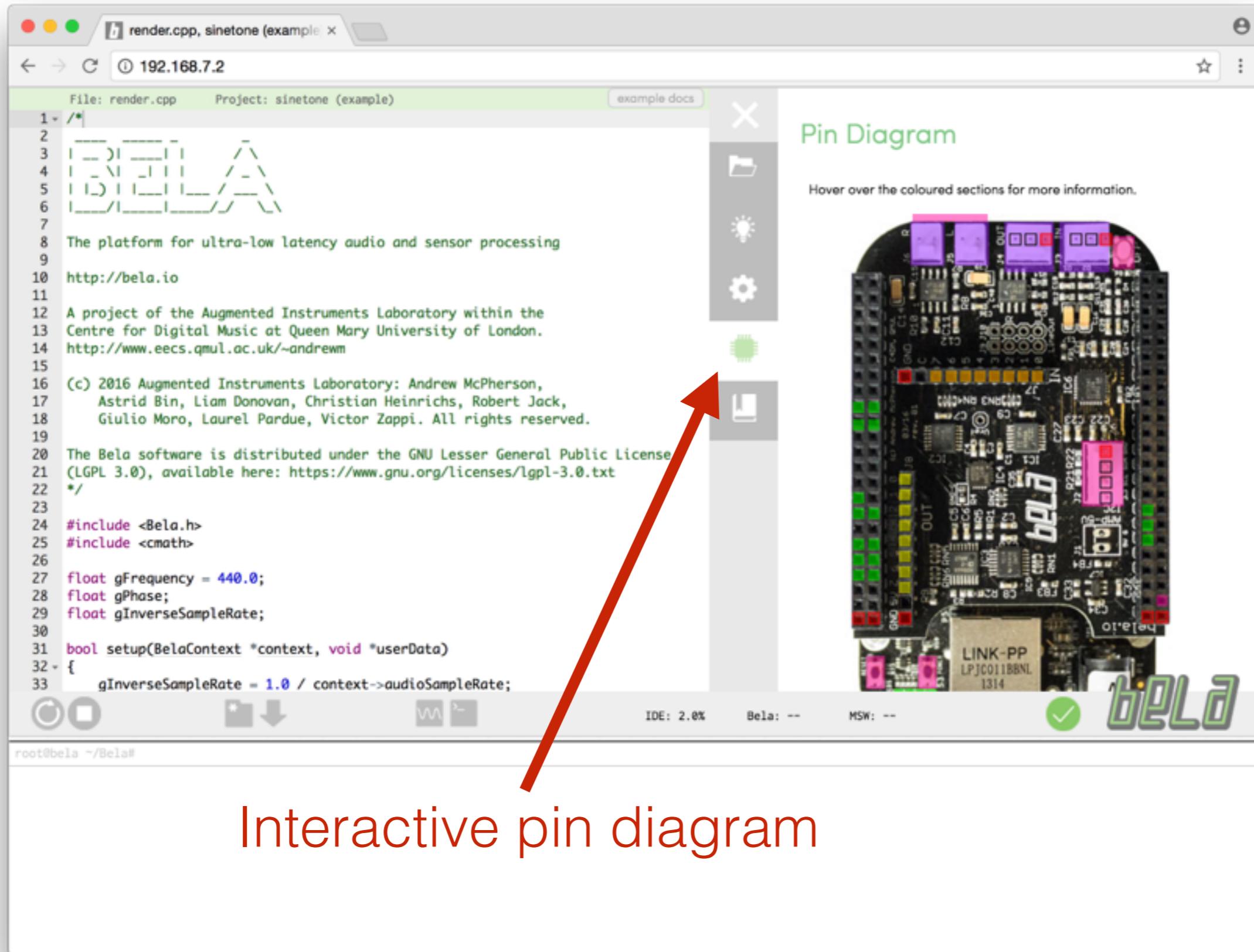
Bring up the Bela IDE: <http://bela.local/>



Bring up the Bela IDE: <http://bela.local/>



Bring up the Bela IDE: <http://bela.local/>



The screenshot shows the Bela IDE interface. On the left, a code editor displays the file `render.cpp` from the project `sinetone (example)`. The code includes comments about the Bela platform, its license (LGPL 3.0), and copyright information. A red arrow points from the text "Interactive pin diagram" at the bottom to the pin diagram panel on the right. The pin diagram panel shows a detailed schematic of the Bela hardware with various pins color-coded and labeled. A green checkmark icon is visible in the bottom right corner of the panel.

```
File: render.cpp Project: sinetone (example) example docs
1 /*
2
3
4
5
6
7
8 The platform for ultra-low latency audio and sensor processing
9
10 http://bela.io
11
12 A project of the Augmented Instruments Laboratory within the
13 Centre for Digital Music at Queen Mary University of London.
14 http://www.eecs.qmul.ac.uk/~andrewm
15
16 (c) 2016 Augmented Instruments Laboratory: Andrew McPherson,
17 Astrid Bin, Liam Donovan, Christian Heinrichs, Robert Jack,
18 Giulio Moro, Laurel Pardue, Victor Zappi. All rights reserved.
19
20 The Bela software is distributed under the GNU Lesser General Public License
21 (LGPL 3.0), available here: https://www.gnu.org/licenses/lgpl-3.0.txt
22 */
23
24 #include <Bela.h>
25 #include <cmath>
26
27 float gFrequency = 440.0;
28 float gPhase;
29 float gInverseSampleRate;
30
31 bool setup(BelaContext *context, void *userData)
32 {
33     gInverseSampleRate = 1.0 / context->audioSampleRate;
```

IDE: 2.0% Bela: -- MSW: --

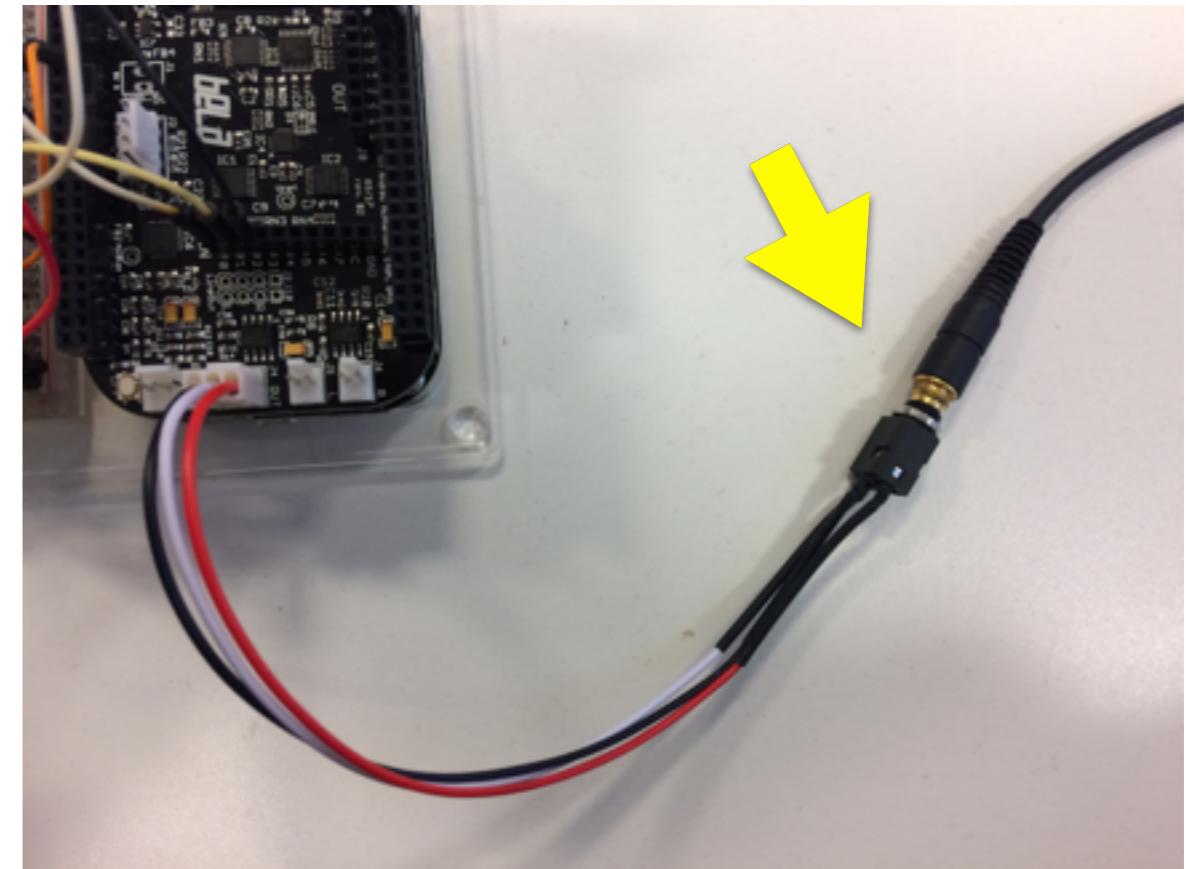
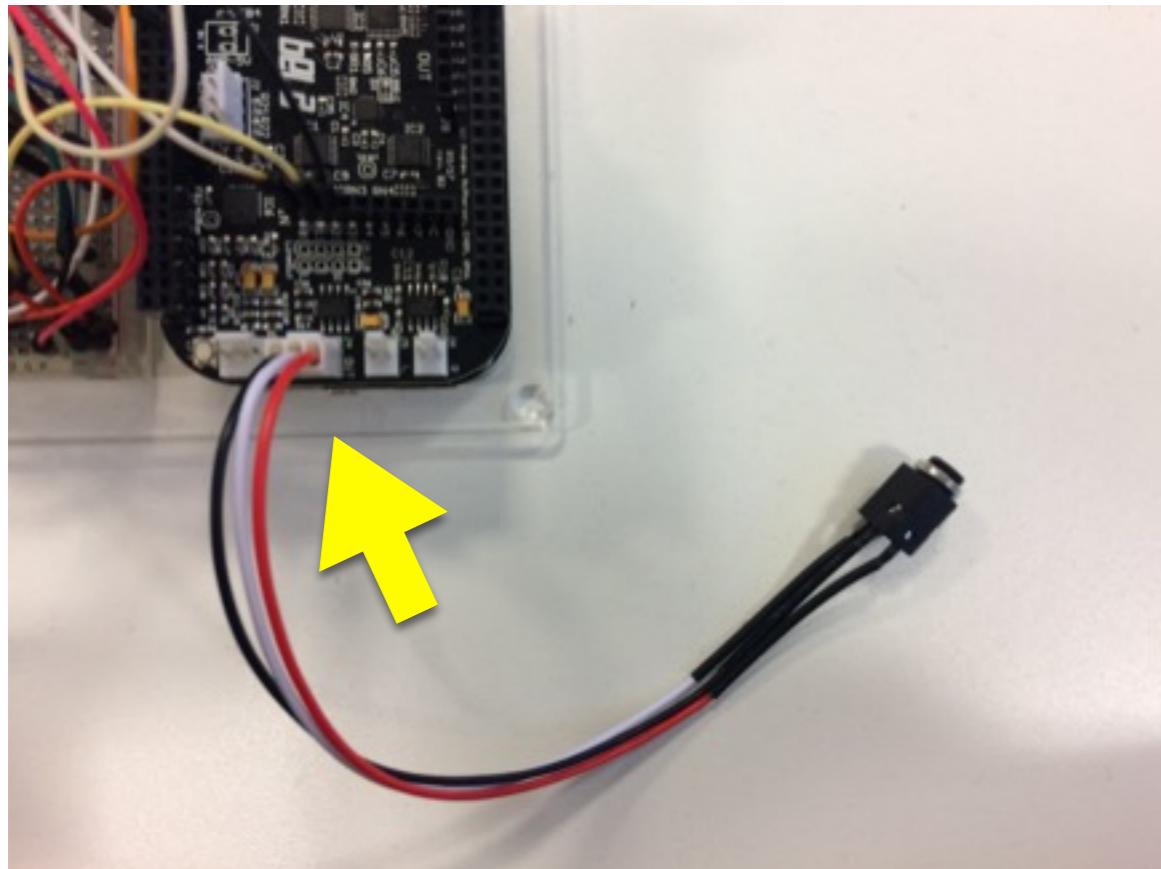
Interactive pin diagram

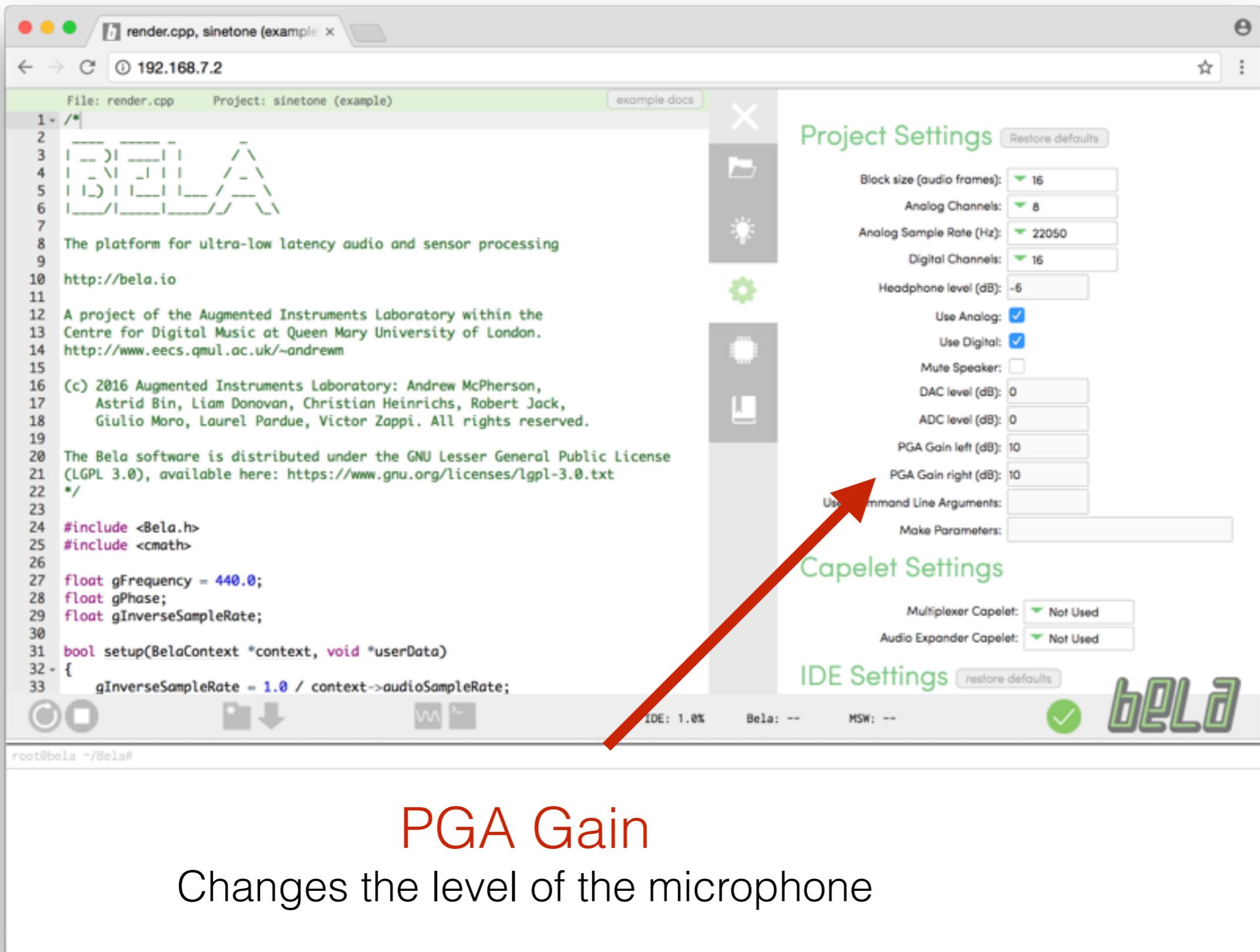
A C++ project is now running **on the board**
not on your personal computer

To listen:

Plug the audio adapter cable into
the 2nd audio out molex connector

Plug headphones into the
audio adapter cable





C++ API

- In `render.cpp`....
- Three main functions:
- `setup()`
*runs once at the beginning, before audio starts
gives channel and sample rate info*
- `render()`
*called repeatedly by Bela system ("callback")
passes input and output buffers for audio and sensors*
- `cleanup()`
*runs once at end
release any resources you have used*
- Code docs available in sidebar of IDE, or at docs.bela.io

Hello world: sinetone

```
#include <Bela.h>
#include <cmath>

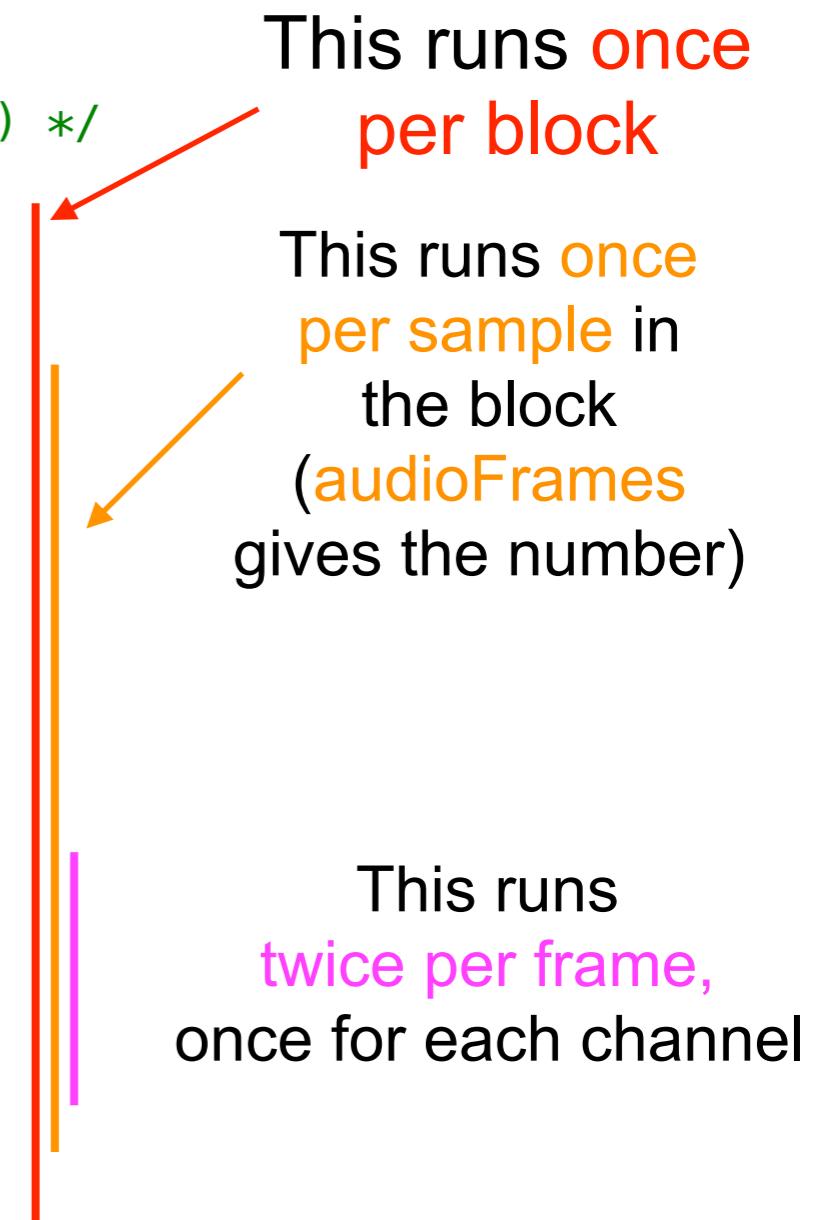
float gPhase = 0.0; /* Phase of the oscillator (global variable) */

void render(BelaContext *context, void *userData)
{
    /* Iterate over the number of audio frames */
    for(unsigned int n = 0; n < context->audioFrames; n++) {
        /* Calculate the output sample based on the phase */
        float out = 0.8 * sinf(gPhase);

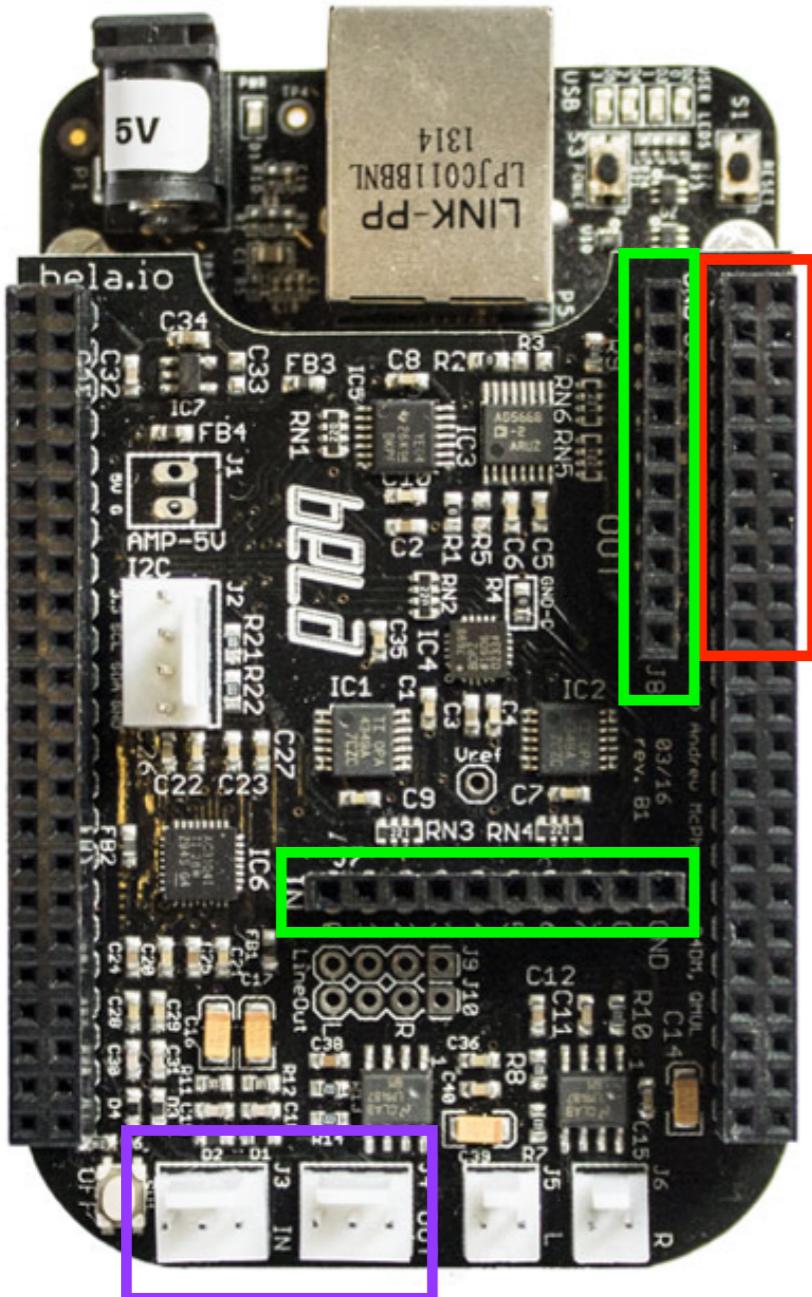
        /* Update the phase according to the frequency */
        gPhase += 2.0 * M_PI * gFrequency * gInverseSampleRate;
        if(gPhase > 2.0 * M_PI)
            gPhase -= 2.0 * M_PI;

        for(unsigned int channel = 0;
            channel < context->audioOutChannels; channel++) {
            /* Store the output in every audio channel */
            audioWrite(context, n, channel, out);
        }
    }
}
```

write to buffer of interleaved audio data,
specifying a **frame**, a **channel** and a **value**



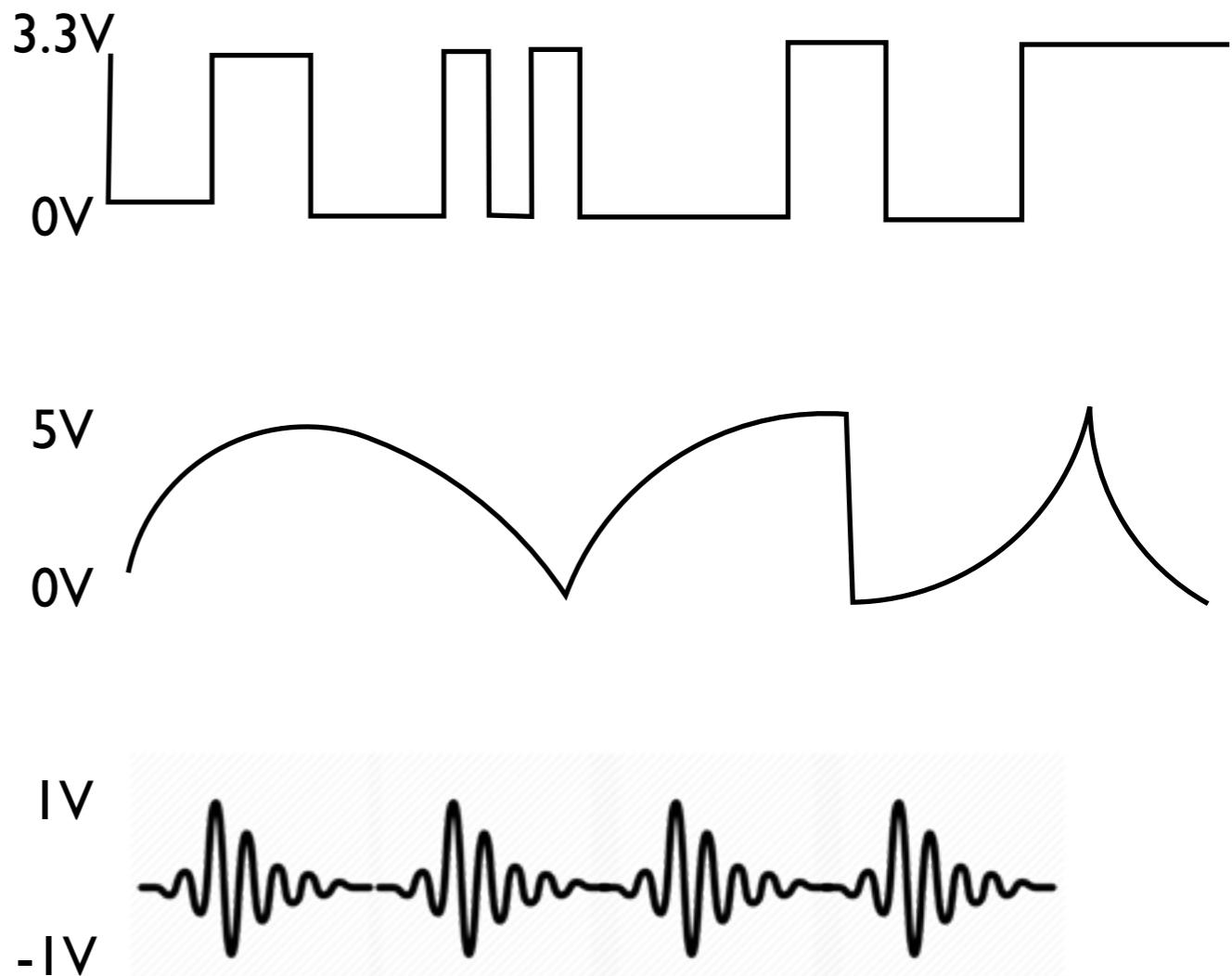
Bela hardware



Digital
in/out

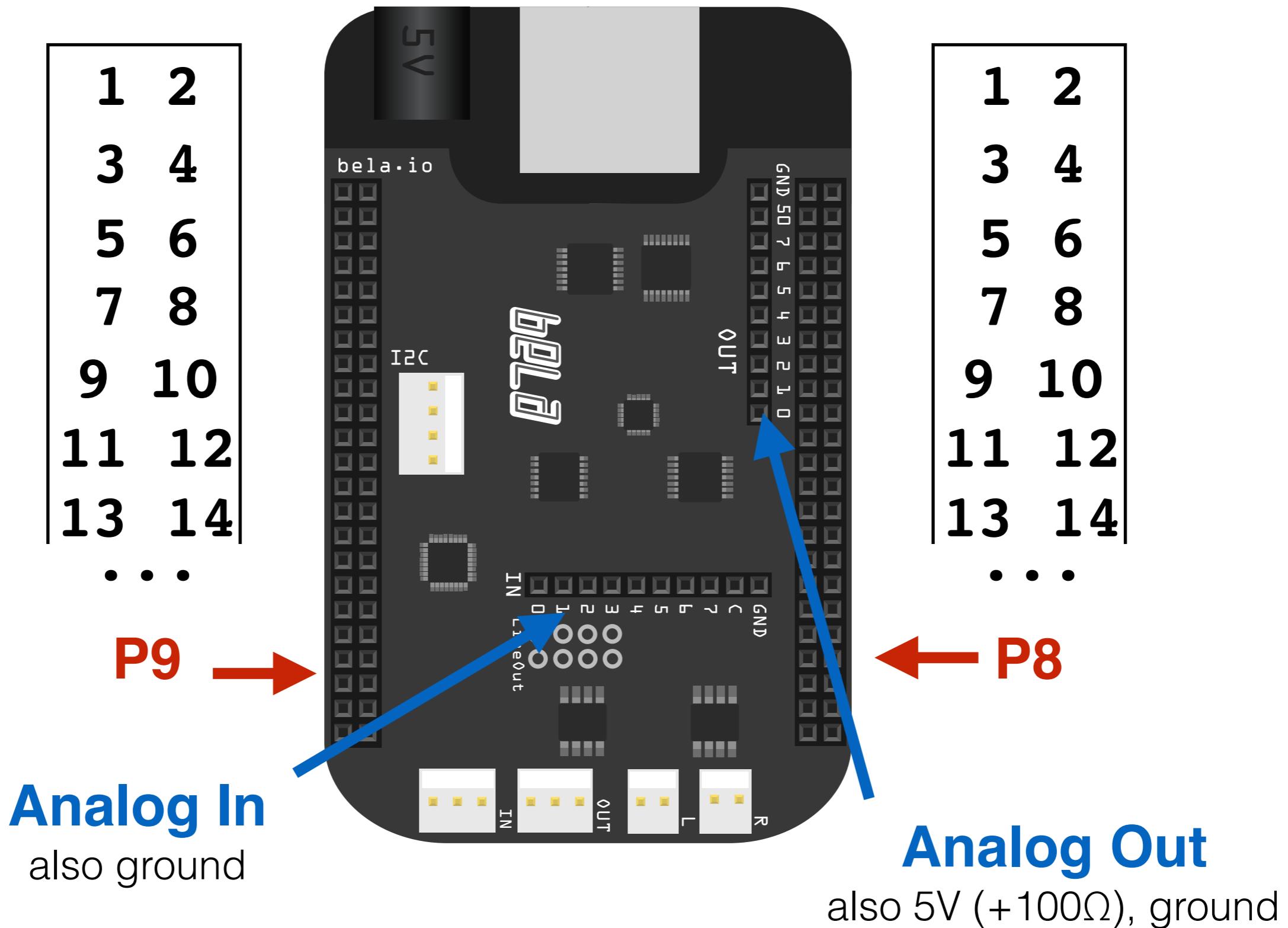
Analog
in/out

Audio
in/out



Bela pin numbering

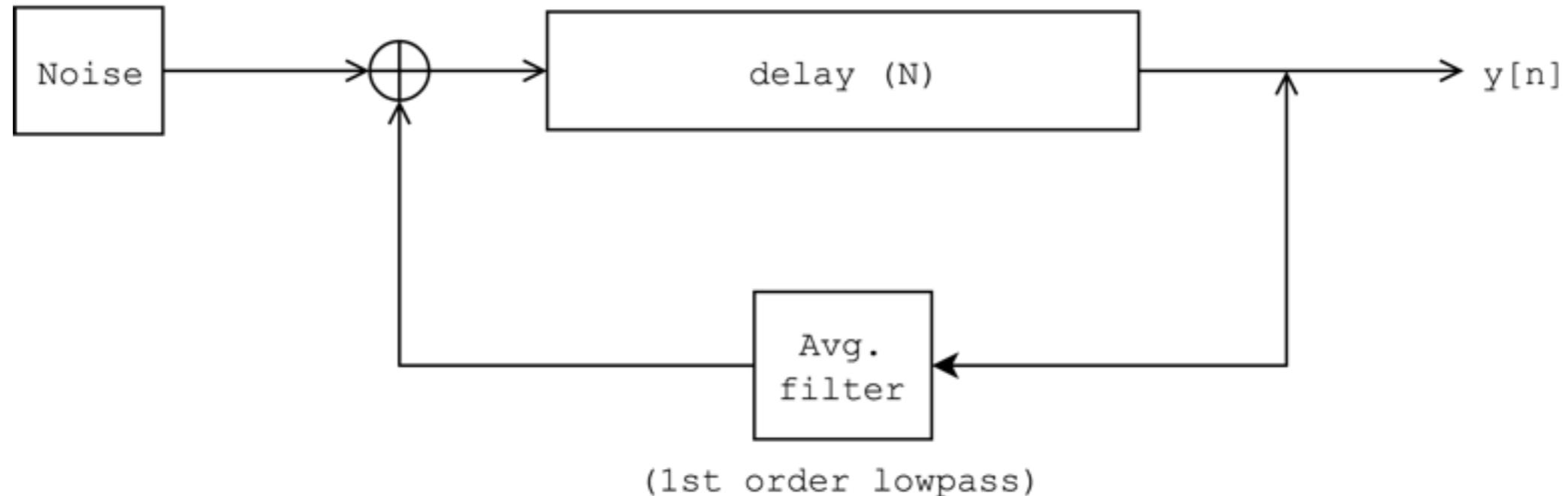
Important: **NEVER** use **5V** with digital I/Os!



Keep everything on
your breadboard!

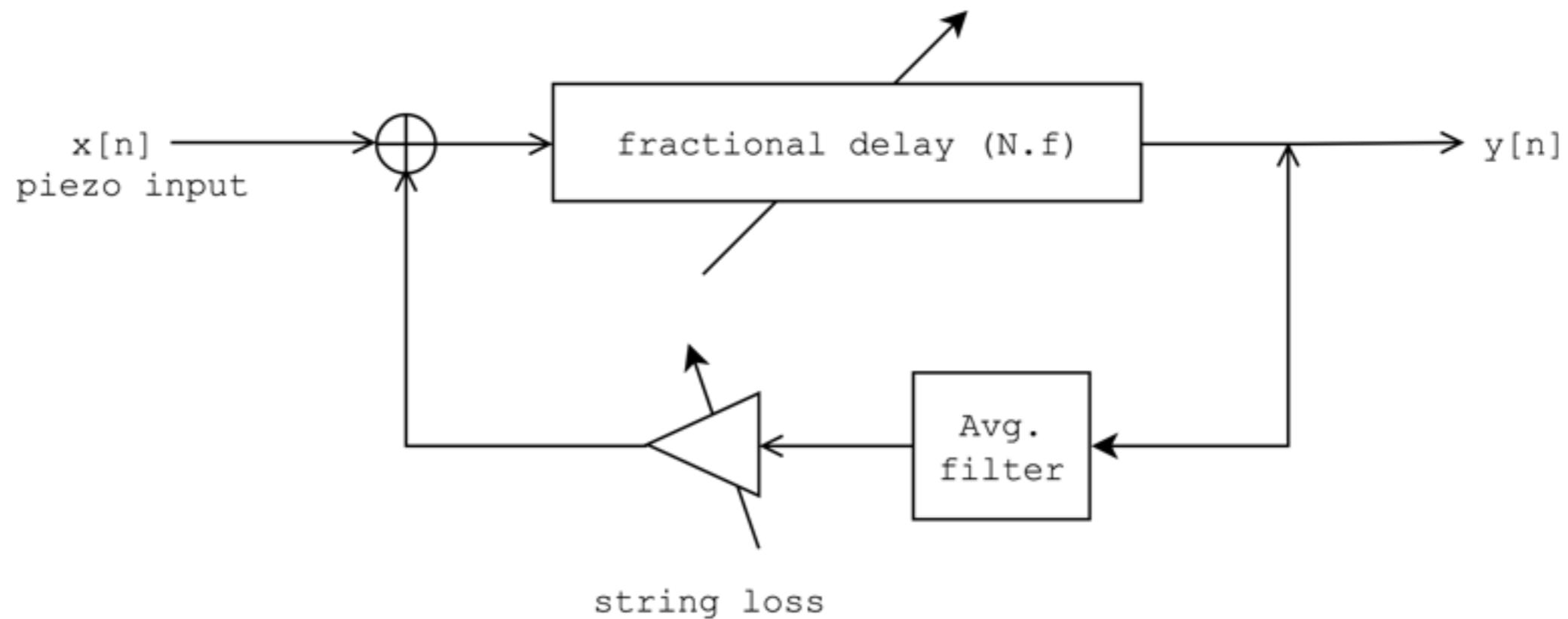
We will be using all sensors
through the workshop

The Karplus-Strong algorithm



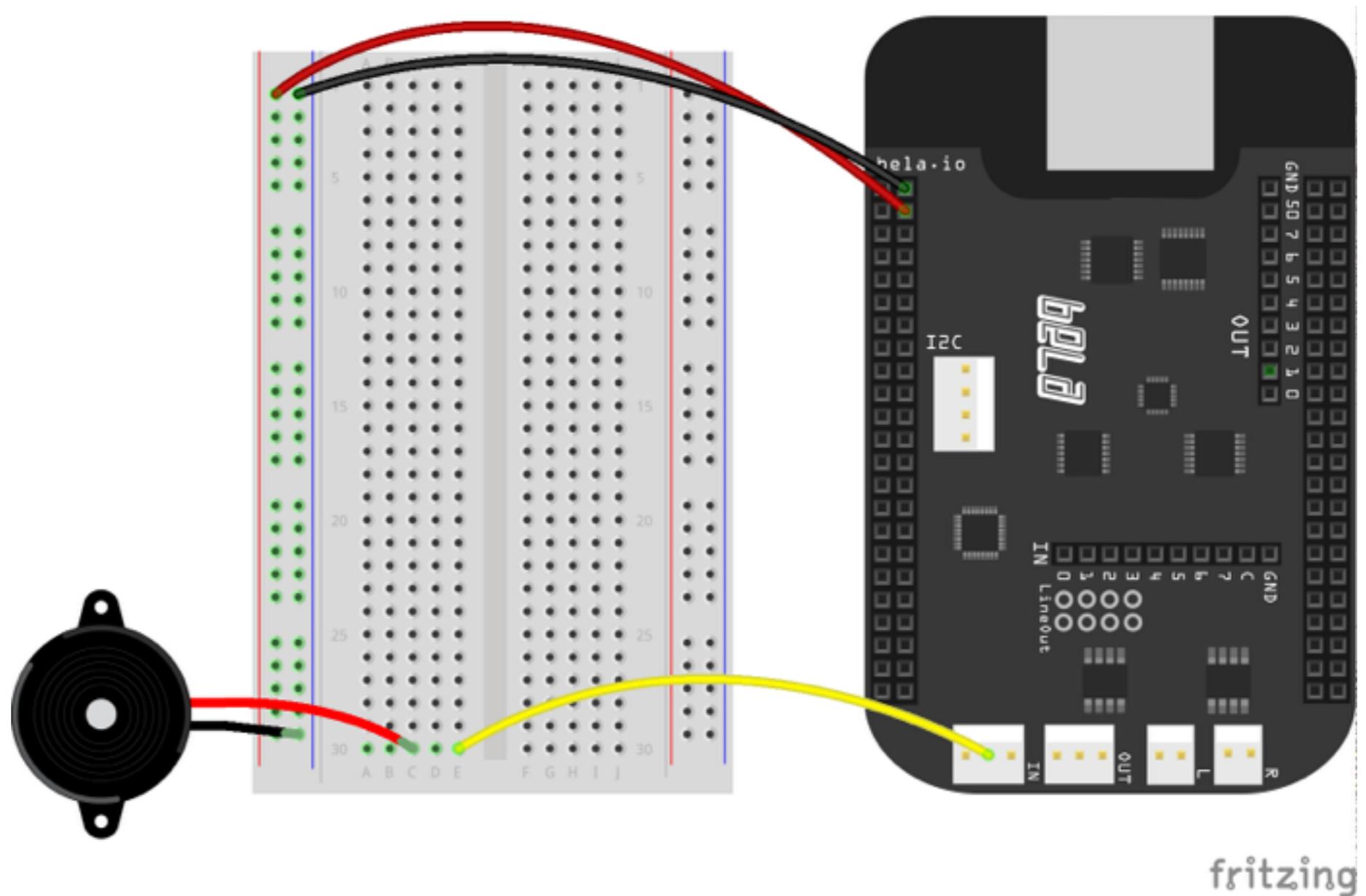
- [1] Karplus, K. and Strong, A., 1983. Digital synthesis of plucked-string and drum timbres. *Computer Music Journal*, 7(2), pp.43-55.
- [2] Jaffe, D.A. and Smith, J.O., 1983. Extensions of the Karplus-Strong plucked-string algorithm. *Computer Music Journal*, 7(2), pp.56-69.
- [3] Karjalainen, M., Välimäki, V. and Tolonen, T., 1998. Plucked-string models: From the Karplus-Strong algorithm to digital waveguides and beyond. *Computer Music Journal*, 22(3), pp.17-32.
- [3] <http://amid.fish/karplus-strong>

Karplus-Strong(ish)



Karplus-Strong(ish)

Step 0



examples/00-Workshop/karplus-strong

Karplus-Strong(ish)

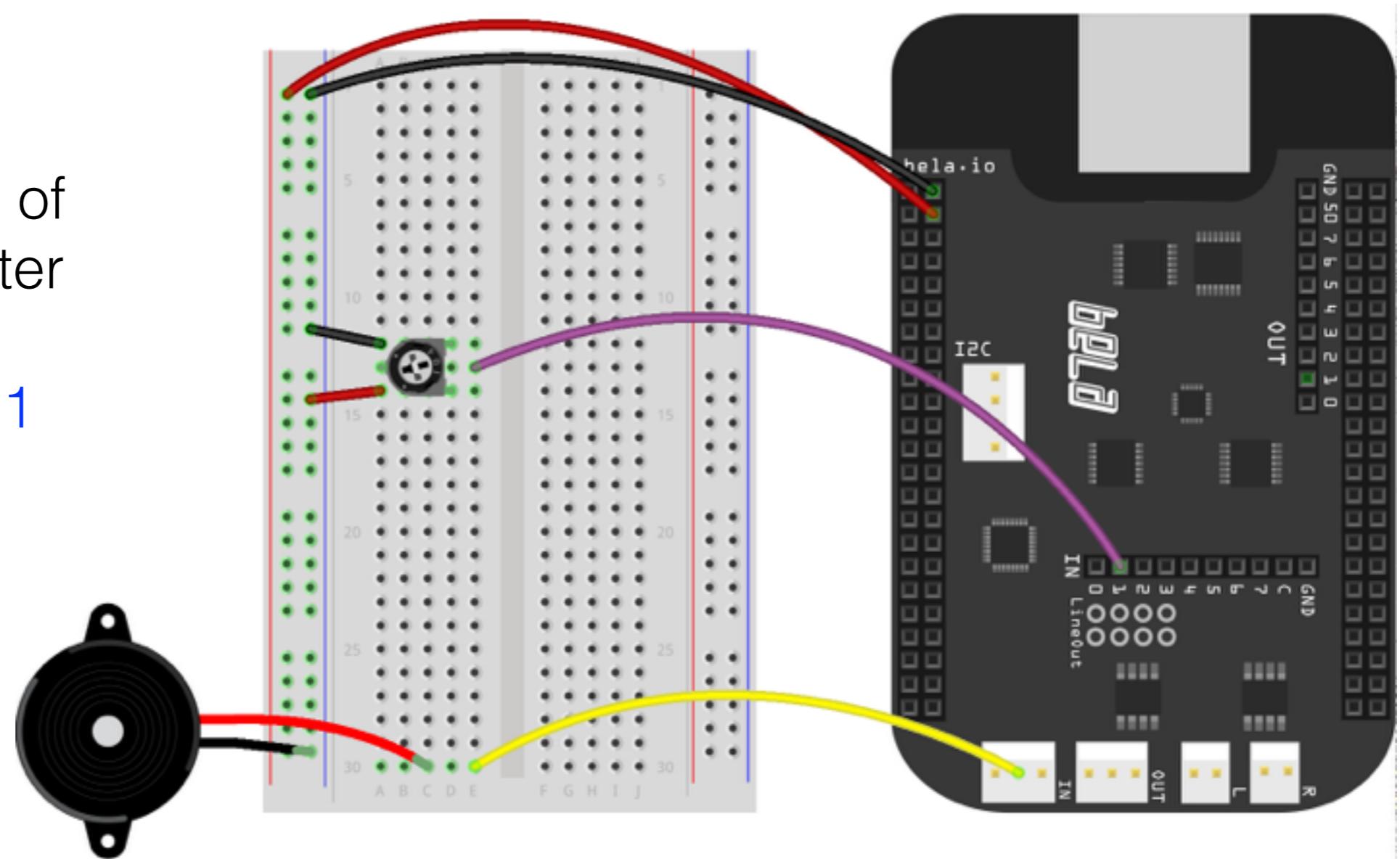
Step 0

- Piezo connected on audio channel 0
- Tap the piezo and listen to K-S being triggered
- Open scope and look at piezo input and K-S output
- Adjust input gain to adjust clipping

Karplus-Strong(ish)

Step I

Middle wire of
potentiometer
goes to
Analog In 1



fritzing

Karplus-Strong(ish)

Step I

- Potentiometer connected to analog input 1
- Add code to render function:

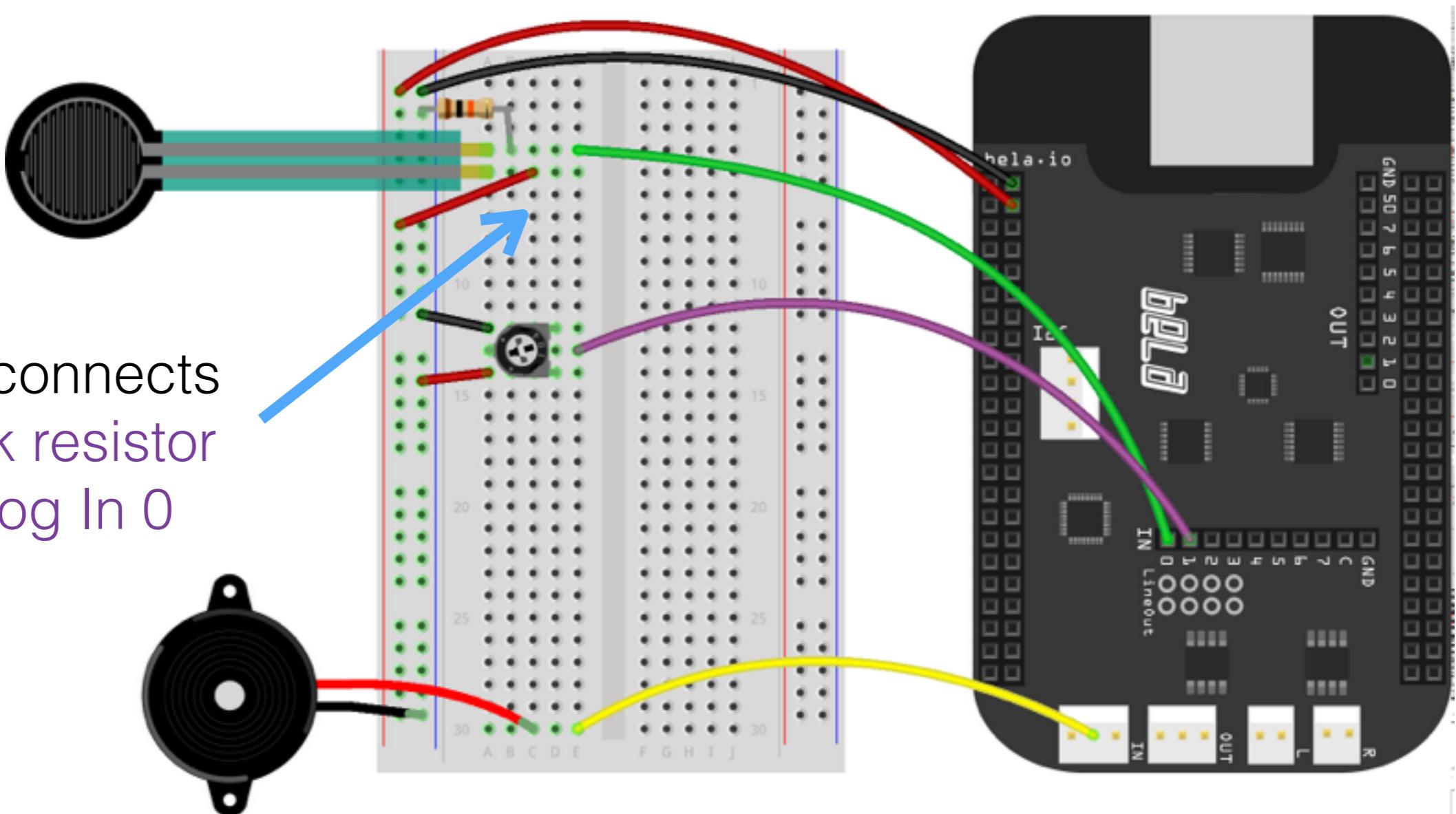
```
// read analog inputs (at audio rate) and update frequency  
  
float potVal = analogRead(context, n, gPotChannel);  
  
float frequency = map(potVal, 0, gAnalogInputRange, gFreqRange[0], gFreqRange[1]);  
  
gPiezoString.setFrequency(frequency);
```

- Use pot to adjust frequency of K-S

Karplus-Strong(ish)

Step 2

Same row connects to FSR, 10k resistor and Analog In 0



fritzing

Karplus-Strong(ish)

Step 2

- FSR connected to analog input 0
- Add code to render function:

```
// read analog inputs (at audio rate) and update loss factor  
  
float fsrVal = analogRead(context, n, gFsrChannel);  
  
float lossFactor = map(fsrVal, gAnalogInputRange, 0, gLossFactorRange[0],  
gLossFactorRange[1]);  
  
gPiezoString.setLossFactor(lossFactor);
```

- Use FSR to adjust loss factor
- Is the FSR response suitable for this application?

Karplus-Strong(ish)

Step 2

- Log FSR to scope:
 - update number of channels in setup()
 - add fsrVal to gScope.log()
- Let's clip the bottom range of the FSR, change/update code:

```
fsrVal = constrain(fsrVal, gFsrRange[0], gFsrRange[1]);  
  
float lossFactor = map(fsrVal, gFsrRange[1], gFsrRange[0], gLossFactorRange[0],  
gLossFactorRange[1]);
```

- Log lossFactor to scope

Karplus-Strong(ish)

Step 3

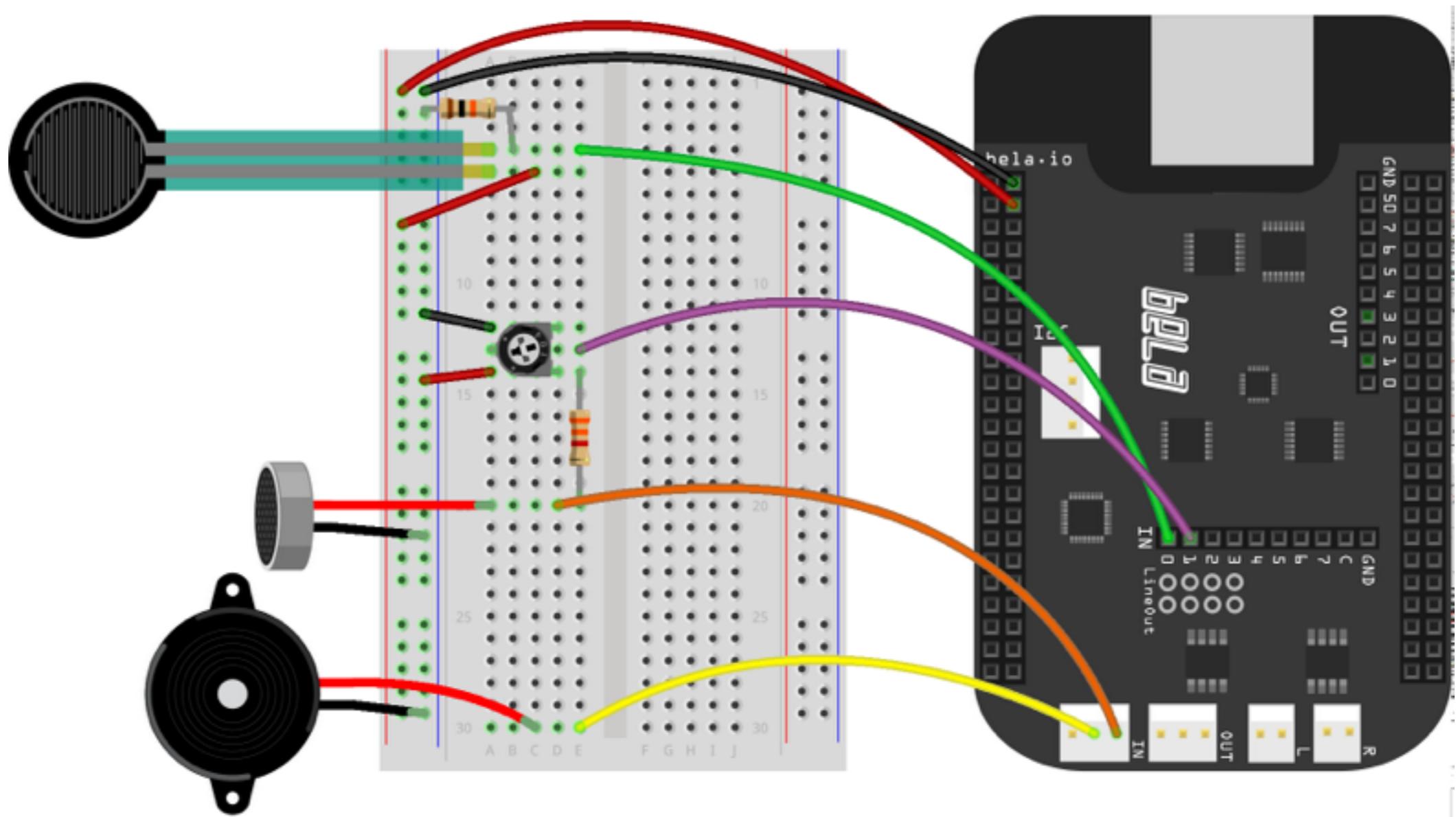
- Is this the response we want from the FSR?
- Can it be better?
- Try log mapping:

```
float fsrLog;  
  
fsrLog = logMap(fsrVal, gFsrRange[1], gFsrRange[0], gLossFactorRange[0],  
gLossFactorRange[1]);  
  
float lossFactor = fsrLog;
```

- Log mapped FSR (fsrLog) to scope

Karplus-Strong(ish)

Step 4



Karplus-Strong(ish)

Step 4

- Add/replace microphone input (audio channel 1)

```
...setup()...

    gMicString.setup(context->audioSampleRate, gFreqRange[0], 432.f *
gFreqRatio);

...render()...

    gMicString.setLossFactor(lossFactor);

    gMicString.setFrequency(frequency * gFreqRatio);

    ...

    float micInput = audioRead(context, n, gMicChannel);

    float micStringOut = gMicString.process(micInput);

    ...

    audioWrite(context, n, ch, gOutputGain * (piezoStringOut + micStringOut));

    ...
```

Karplus-Strong(ish)

Step 5

- Loss factor can be improved
- Add unable LP filter to feedback loop for more realistic damping
- In KarplusStrong.h, comment out :

```
#define KS_CONSTANT_LOWPASS
```

- In render () :

```
#ifdef KS_CONSTANT_LOWPASS  
....  
#else /* KS_CONSTANT_LOWPASS */  
    fsrLog = logMap(fsrVal, gFsrRange[1], gFsrRange[0], gDampingRange[0],  
    gDampingRange[1]);  
  
    float damping = fsrLog;  
  
    gPiezoString.setDamping(damping);  
  
    gMicString.setDamping(damping);  
  
#endif /* KS_CONSTANT_LOWPASS */
```

Bela + Karplus-Strong Instruments



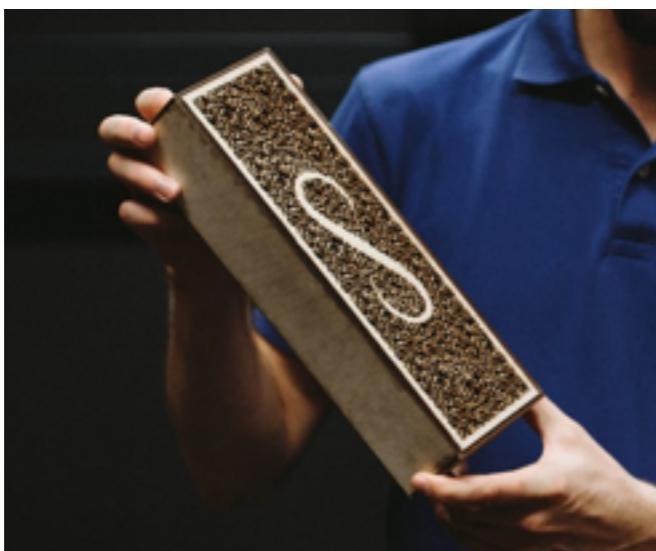
Kalichord Strum

Dan Moses



Strummi

Jacob Harrison, Robert H. Jack



Air Harp

Chris Heinrichs



KSSynth

Rej Poirier

Finding out more

<http://blog.bela.io>



The screenshot shows a web browser window for the Bela Blog at <https://blog.bela.io>. The page has a header with three tabs: 'BLOG' (underlined), 'CATEGORIES', and 'ABOUT BELA'. Below the header, there's a large image of a Bela Mini board. The main content area features two blog posts:

Nov 18, 2018
Live Electronics at the Conservatorium van Amsterdam with Bela Mini
At the Conservatorium van Amsterdam Jos Zwaanenburg has been using a fleet of Bela Minis as part of his teaching on the Master's in Live Electronics. Ultra-low Latency Live Electronics...
[READ MORE](#)

Oct 12, 2018
Prototyping Spatial Audio for VR/AR with Bela Mini
In this post we discuss how Bela can be used to prototype immersive

Finding out more

<http://forum.bela.io>

The screenshot shows the Bela forum homepage with a list of recent discussions. The sidebar on the left contains links for 'All Discussions', 'Following', 'Tags', 'FAQ', 'General', 'Getting Started', 'Interactivity', 'Audio', 'Hardware', 'Software', 'Show and Tell', 'Forum', 'CTAG-ALSA', 'Solved', and 'Add to wiki'. The main content area displays ten discussions:

- Heavy, Pd and Enzienaudio** by Jukkapolka (replied an hour ago) - Software, Pure Data, 1 reply
- Monome Grid & Bela** by padenot (replied 2 hours ago) - Interactivity, 1 reply
- Scope sliders initialization** by stephanelesolinne (replied a day ago) - Software, 3 replies
- How to drive a LRA by bela board?** by giuliomoro (replied 2 days ago) - Audio, 3 replies
- high-performance-mode** by giuliomoro (replied 2 days ago) - General, 4 replies
- Bela as a plug and play USB Audio Soundcard Device?** by Gladgrif (replied 3 days ago) - Audio, 6 replies
- Bela out to XLR?** by giuliomoro (replied 3 days ago) - Hardware, 2 replies
- Reading multiple samples and CPU overload.** by phevoso (replied 3 days ago) - Software, Pure Data, 2 replies
- Problem with scope after upgrade to last version** by giuliomoro (replied 3 days ago) - Software, IDE, 24 replies

Finding out more

<https://github.com/BelaPlatform/Bela/wiki>

The screenshot shows the GitHub Wiki page for the Bela repository. The page title is "Home · BelaPlatform/Bela Wiki". The header includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and user statistics: 37 unwatched, 195 stars, and 44 forks. Below the header, there are links for Code, Issues (166), Pull requests (1), Projects (2), Wiki (selected), Insights, and Settings. The main content area features a large, stylized Bela logo composed of thick black outlines. To the right of the logo is a sidebar with a "Pages 56" section containing links to "Getting started with Bela", "Bela IDE", "Bela Hardware", "Examples and Tutorials", and "Updating Bela". At the bottom, there is a link to "Clone this wiki locally" with the URL <https://github.com/BelaPl1>.

Home

Robert Jack edited this page on 15 Oct · 90 revisions

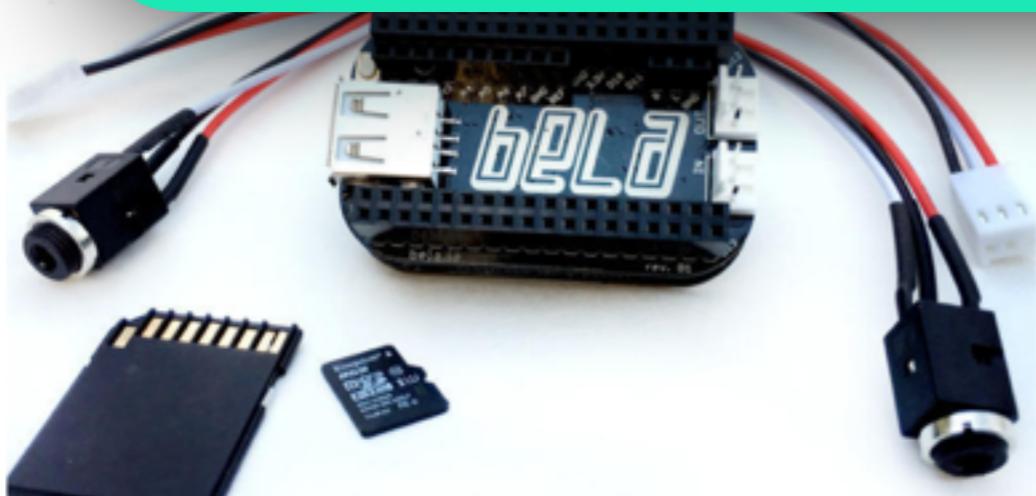
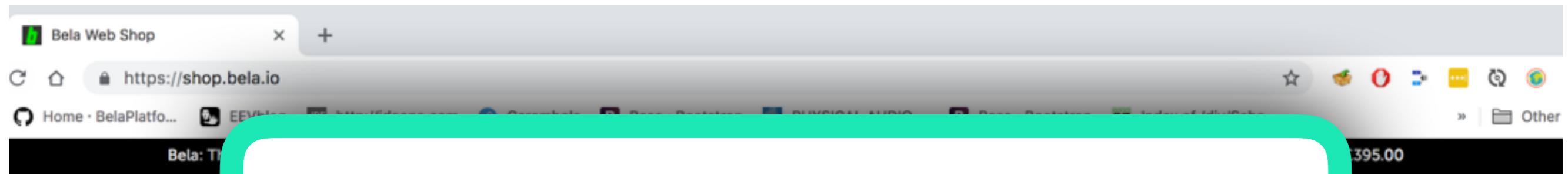
Bela

Introducing Bela

Bela is an open-source embedded computing platform for creating responsive, real-time interactive systems with audio and sensors. It features ultra-low action-sound latency, high-

Where can I get one?

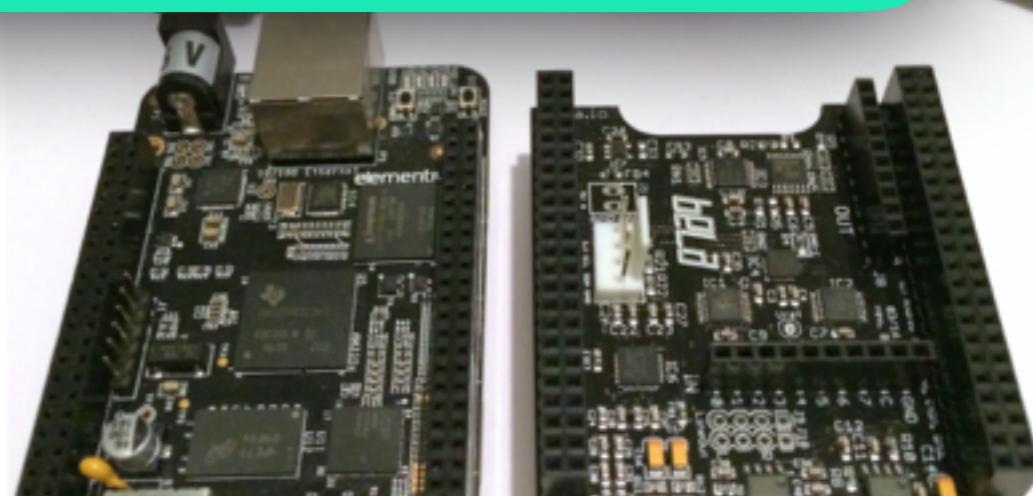
<https://shop.bela.io/>



BELA MINI CAPE AND KITS

The new, smaller Bela Mini cape, based on the PocketBeagle. It has stereo audio input and output, 8 analog inputs, 16 digital I/O.

[SHOP NOW ▶](#)



BELA CAPE AND KITS

The fully-featured Bela cape and kits, based on the BeagleBone Black. It has stereo audio input and output, 8 analog inputs and outputs, 16 digital I/O, two power speaker amplifiers.

[SHOP NOW ▶](#)