

# Using Ambisonics as production format

---

This document assumes basic familiarity with Ambisonic concepts and techniques.

## Introduction

Ardour can be used to create and process Ambisonic content, up to very high orders, thanks to its flexible routing concept with an arbitrary number of channels per route.

When you want to create an Ambisonic mix, your Master bus, and any group busses you might want to use, become B-Format buses, with 4 channels for first-order, 9 for second-order, 16 for third-order, and so on. Or you could use intermediate channel counts for mixed-order setups. It is important to note that Ardour does not care what kind of data you send over those buses. The only Ardour components which do care are the panners, which currently only support discrete pair-wise panning or some arcane multi-channel scheme, and hence must be bypassed.

Keep in mind it's the panners which decide what each of the channels in your master bus "means": They define "Left" and "Right", or "W", "X", "Y", "Z" and so on.

## Setting up for discrete source panning

Instead of the built-in panners, you can use the [AMB LADSPA plugin set](#)<sup>1</sup> by Fons Adriaensen, which provides spherical panners in 1st, 2nd, and 3rd order flavours. They use the Furse-Malham convention for channel ordering (W XYZ RSTUV KLMOPQ) and MaxN normalisation (i.e. each channel is normalized to fit between 0..1 for input at full scale). Each panner has two knobs, namely azimuth (horizontal angle, from -180 to 180) and elevation (vertical angle, from -90 to 90). Like all plugin parameters, these can be automated.

Before you start, some preparation is necessary:

- Create a new session with an appropriate multi-channel master bus depending on the order you want to work in.
- Activate the option "Run plugins during recording", so that the panners are active while you're tracking (if you care about proper monitoring).
- Create a mono track, bypass the default panner and add a suitable panning plugin into the post-fader signal stream of each track.
- Optionally, add some other utility plugins such as an EQ and a compressor, and save this track as a route template, from which you can then create as many tracks as you need.

## Integrating natural Ambisonic recordings

The above paragraph assumed that you are panning discrete mono sources around on the Ambisonic sphere. If you have a first or higher-order soundfield microphone, you can also integrate natural Ambisonic recordings into your mix.

Note that it is generally advisable to record the A-Format of your microphone, i.e. the unprocessed, raw capsule output. That way, you can always fine-tune your B-Format encoding matrix in case anything went wrong at the recording location.

But this also means, that you cannot feed your A-Format track directly into the Master bus. Instead, you have to encode it to B-Format first. Use the tool that came with your microphone to accomplish this task. Users of the CoreSound TetraMic can patch the TetraProc processor into a post-fader 4channel insert and connect it in realtime via JACK. Other vendors might provide VST plugins for the same task. YMMV. As a last resort, create an offline B-Format rendering and import it into a suitable track of your session.

---

1. <http://kokkinizita.net/linuxaudio/downloads/index.html>

Of course, now your Master bus will output B-Format, which cannot be fed to your monitor speakers directly. Instead, you will need an Ambisonic decoder to match your speaker layout.

For a real ambisonic setup, use Fons Adriaensen's [ambdec](#)<sup>2</sup>. Be sure to check "Furse-Malham" input normalization, and connect the appropriate ports of your Ardour Master bus to ambdec's inputs. In turn, ambdec's outputs will feed your speakers.

If you need to render to stereo, you can use UHJ encoding, which will give a very pleasurable and reasonable two-channel rendering, but it is not suitable for serious surround monitoring. An UHJ encoding configuration example comes with [jconv](#)<sup>3</sup>, which is a very handy convolution engine also useful for high-end reverb sounds.

If your target format is 5.1, you can still use Ambisonics as a production format, and extract a 5.0 feed afterwards, again using ambdec with a specially optimized decoding configuration (the LFE channel is not suitable for music anyways and should be handled separately, as you would in a normal 5.1 session). To obtain high channel separation in ITU 5.1 as required by the movie industry, it's advisable to use at least third-order Ambisonics. For natural, coherent sound fields, second order is sufficient.

When working in higher orders, it is highly advisable to set up some decoders for lower orders in parallel, so that you can quickly check your mixes for people with less spiffy listening rigs. A setup that works well is to have the master bus feed a couple of other buses, one for each order you want to be able to monitor, and then mute/unmute those as needed. Each of those buses is routed into a separate ambdec instance (or, in the case of UHJ, a jconv process).

## A note on Ambisonic channel order and normalization

The traditional Furse-Malham format is an extension of Michael Gerzon et al.'s design from the 1970s. Back then, limitations of the existing magnetic recording media were the dominant engineering constraint. Hence, it was decided to drop the W channel by 3dB, and clamp each of the other channels to a defined maximum (*MaxN normalisation*). This made no sense from a mathematical point of view (and in fact needed to be corrected before decoding), but it ensured the best S/N performance on the tape machines of the era.

With the extension of Ambisonics to higher orders, the crookedness in channel ordering became evident (easily seen when you look at the shapes of the spherical harmonics and their behaviour with respect to rotations). Moreover, with modern digital media, the historic normalisation compromises have lost their justification and are becoming increasingly cumbersome. Hence, a new, inherently logical, mathematically convenient, encoding scheme has been proposed: [ACN](#)<sup>4</sup>, or Ambisonic Channel Number. Similarly, current implementations favor SN3D or N3D normalisation over the old MaxN scheme, because it maintains meaningful level relations between the components.

It is to be expected that future panner implementations will move to this new scheme, and you will have to be aware of this possible complication. Many Ambisonics research facilities are already moving their in-house formats to ACN/SN3D.

For back- and forwards compatibility, trivial adaptor plugins can be used.

## Further reading:

- ["Using Ambisonics with Ardour"](#)<sup>5</sup> from ardour.org (2009)
- ["Ardour and Ambisonics - A FLOSS approach to the next generation of sound spatialisation"](#)<sup>6</sup> (09/2009), from eContact! magazine.
- an ambisonic production [case study](#)<sup>7</sup> ([slides](#)<sup>8</sup>, [video](#)<sup>9</sup>)
- ["Ambisonics"](#)<sup>10</sup> in Wikipedia

---

2. <http://kokkinizita.net/linuxaudio/downloads/index.html>

3. <http://kokkinizita.net/linuxaudio/downloads/index.html>

4. <http://ambisonics.ch/standards/channels/>

- ["UHJ"](#)<sup>11</sup> in Wikipedia