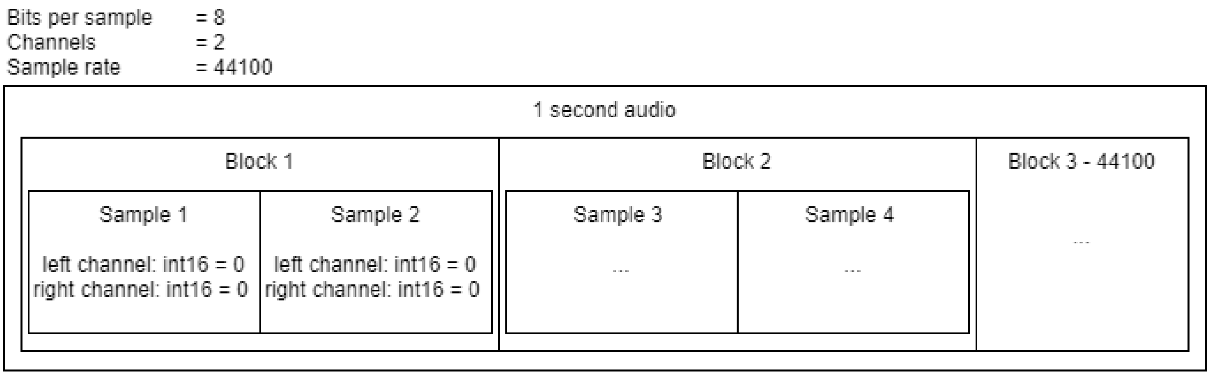
PCM sample rate conversion

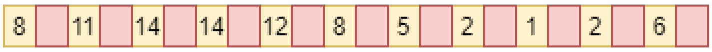
PCM is the native data representation used in many of the available hardware to stream audio. It looks a bit like the following graphic.

Channels is a synonym to audio speakers in this context.
The sample rate is the amount of samples per second for each channel.

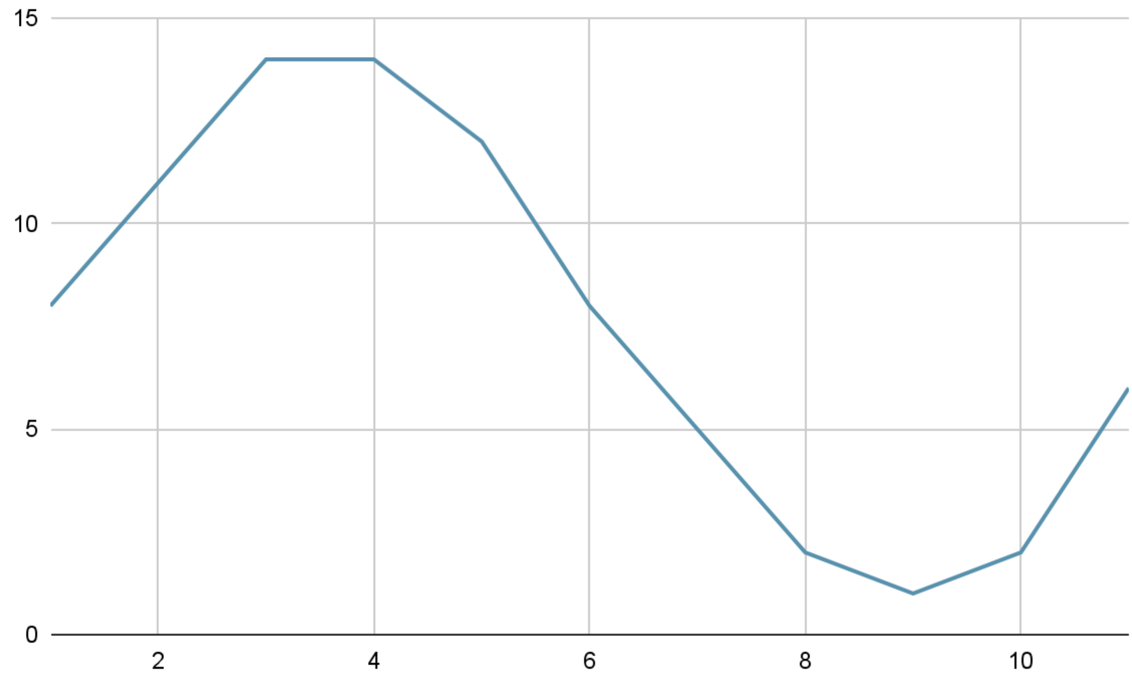


[This](#) is a good site containing more information about PCM and the wave file format.

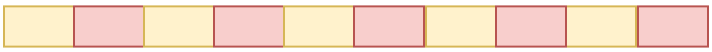
One second of pcm audio data with the imaginary sample rate of 11 and 2 channels looks in memory somewhat like the following graphic. Each square is the size of *bits per second* and represents a value on the y axis of a digital wave. There is a wave per channel and in memory the data is interleaved. In this example the yellow squares are the left channel and the red ones the right channel. In total there are 22 squares. This is 11 for the amount of samples per second times 2 for the amount of channels.



When we take a look just at the left channel and draw the digital wave this represents we get something like this. Which is an approximation of a really ugly sine wave.



A second of audio with the sample rate of 5 and 2 channels looks similar to this. (The size of the blocks is not to scale with the amount of memory needed but with the length of audio each block represents, each block is still the size of *bits per second* and the length of the total audio is still one second)



If we want to convert from one sample rate to another we can either interpolate the nearest values or we can insert values representing silence. This has to be done on a per channel basis to make sure they are not mixed with each other.

If we want to interpolate the data we could group the samples with the same color together. The smaller blocks are colored with the same color as the nearest bigger block of the same channel.

In most real world cases the audio we have is not a multiple of seconds but can also be a fraction of that. If we have audio with a sample rate of 11 and there are 15 blocks of data the length is 1363.64~ ms.

*1363.64~ms = 1000ms / 11 * 15*
11 = sample rate / blocks per second
15 = blocks

If we have audio with a sample rate of 5 it would need 6.82~ blocks to get the same length.

6.82 = 1363.64~ms / (1000ms / 5)
1363.64~ms = length of source audio
5 = target sample rate

As we can only have full blocks of audio this means a conversion from the sample rate 11 to 5 would result in audio with the length of either 1200ms (6 blocks) or with the length of 1400ms (7 blocks).

The following function shows the resulting length of a new audio file when choosing the larger target file size. If you want the smaller size you can replace the ceiling function ($\lceil x \rceil$) with the floor function ($\lfloor x \rfloor$).

oldLengthInSeconds = oldByteCount / bytesPerSample / channels / oldSampleRate
*newBlockCount = ⌈ oldLengthInSeconds * newSampleRate ⌉*
newLengthInSeconds = newBlockCount / newSampleRate

With this we have all the information to design a sample rate conversion algorithm.