



ULCC, Ultra Low Complexity Coding for audio

Since the introduction of digital audio, with the release of the Compact Disc standard at 16-bit and 44.1 KHz sample rate, people have wanted to compress digital audio signals for a variety of reasons:

- to increase recording time
- to transmit or exchange audio over the internet
- to transmit audio over a wireless connection
- Etc

Various solutions are currently available, all of which do an excellent job in compressing audio for the purposes listed above. The most widely used compression is mp1, of which there are several types, known as "layers"

- layer 1
- layer 2, widely used in broadcast
- layer 3, widely used on the internet, also known as mp3

Other types are:

- AAC, used increasingly for video and high quality audio.
- SBC, used for bluetooth
- ATRAC, used for Sony minidisc
- APTX, used for high quality audio

These codecs are relatively complex algorithms - some more so than others. The more complex codecs are:

MP1, AAC and APTX

Less complex ones are SBC and ATRAC

This complexity leads to several major disadvantages:

1. Encoding and (to a lesser extent) decoding require considerable computing power.
2. This results in greater consumption of power.
3. Application is limited to products with the available computing power.
4. Some compression types are based on patents from various manufacturers, making licensing more expensive.
5. Implementation of the codec into different platforms is very complex, and requires so much manpower that product innovation can be hampered by development costs, or by the upfront licensing fees payable to the third parties that have already made the implementation.



In this respect, ULCC is believed to be a perfect solution, addressing the above disadvantages in the following ways:

1. Encoding uses <30 instructions per sample, and is entirely integer/number based. Decoding uses <10 instructions per sample and is also integer/number based.
2. This results in little extra current consumption.
3. ULCC can for instance be implemented on a microprocessor with 4 MHz clock and 48 KHz sample rate, so no additional processing is needed.
4. The current basic algorithm has a single inventor.
5. Implementation on different platforms is extremely easy.

Bitrate

ULCC's bitrate is dependent on the required audio quality

Typical high quality bitrates for ULCC at a 48 KHz sample rate are:

- 16-bit accuracy: 240 kbps per channel, compared to 768 kbps uncompressed.
- 24-bit accuracy: 262 kbps per channel, compared to 1152 kbps uncompressed.

Quality

ULCC is a purely time-based compression algorithm, with the following characteristics:

- Transparent audio quality at bitrates mentioned above.
- Scalable audio quality, from medium to high-end.
- Excellent transient response.
- Wide dynamic range.
- Coding artefacts are only audible in very special circumstances and when audible, are similar to those found in older analog audio equipment.

Special characteristics

- Extremely low coding delay, typically 32 samples.
- When using higher sample rates like 96 kHz, the quality of the compression algorithm improves even more, exponentially.
- Higher bit-depths only increase the bitrate marginally.
- With wireless systems, the power requirements for transmission of ULCC increase compared with those for lower bitrate solutions like mp1 and AAC types. However, the power requirements for encoding and decoding are down to almost nil, with audio quality remaining the same, or better in most cases. It is also possible to lower bit rates to 150 kbps per channel, but this leads to some audible degradation in audio quality. It depends on the application if this degradation is acceptable or not.

Applications

- Portable audio players. Using ULCC dramatically improves battery life.
- Wireless audio transmission. Using ULCC dramatically improves battery life.
- Networked audio. Reduced bitrates improve stability, compared with the use of uncompressed audio.
- Multi-channel audio recording. More channels can be recorded using the same storage write speeds.
- Etc.