A Presentation On:

flac

free lossless audio codec

By

Corey Kereliuk

Presentation Overview

Part I: motivation behind the flac

Part II: description of the flac

Part III: comparison to other lossless audio standards

Part I: motivation behind the flac

Primary goal:

- Develop an open, patent-free standard for lossless audio compression/decompression

Secondary goals:

- Achieve a good compression ratio
- Make encoding/decoding fast
- Include a multi-platform reference implementation and documented API
- Include metadata
- Make stream-able, seek-able
- Make standard extensible

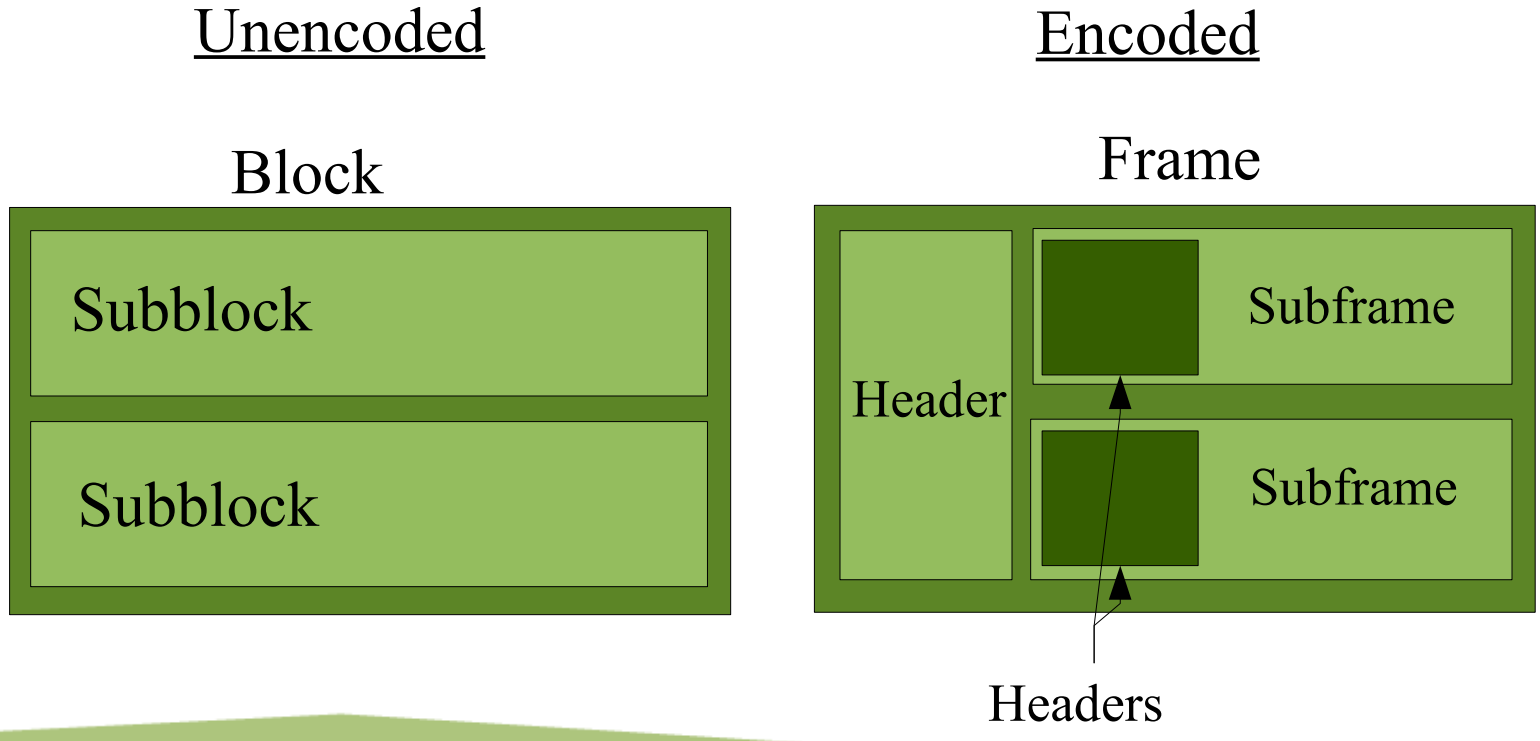
Part II: description of the flac

The architectural components of flac are:

- Blocking
- Inter-channel Decorrelation
- Prediction
- Residual coding

Blocking

- Audio stream is broken up into blocks for encoding (size is variable)
- Block size has a direct effect on the compression ratio (small size = more overhead, large size = worse prediction)



Stream Format

- A flac stream consists of a "fLaC" marker at the beginning of the file, followed by a STREAMINFO metadata block
- flac can support up to 128 different kinds of metadata blocks
- Currently defined metadata blocks are:
  - STREAMINFO (sample rate, channels, etc)
  - PADDING (reserve space to write post-encoding)
  - SEEKTABLE (to help reduce seek times)
  - VORBIS_COMMENT (for tagging files with artist, song, etc)
  - CUESHEET (Red Book CD track data)
  - PICTURE (cover art)
- Each frame header contains a sync code, frame size, frame number or sample number (for seeking), sample rate, number of channels, and an 8-bit CRC
- Each subframe header contains contains information about how that channel was encoded (eg. 4^{th} order linear predictor, w/ run-length coding of the residual)

Interchannel Decorrelation

- There is often some correlation between the different channels in a multi-channel audio signal
- Modes of operation:
  - Independent: code left (L) and right (R) channels independently
  - Mid-side: code mid (L+R)/2 and side (L-R) channels
  - Left-side: code (L) and (L-R)
  - Right-side: code (R) and (L-R)
- The flac encoder will choose the best representation on a frame-by-frame basis

Prediction

- 4 Modes of Modeling the input signal:
  - Verbatim: the predicted signal is zero, and the residual (which is the signal itself) is transmitted (no compression)
  - Constant: Predictor used for silent sections, signal is run-length encoded
  - Fixed Linear Predictor: 4^{th} order linear predictor
  - FIR Linear Prediction: up to 32^{nd} order FIR linear prediction

Linear Prediction

- In LPC the signal is predicted as a linear combination of its past outputs:

$$s'[n] = \sum_{k=1}^{k=p} a_k s[n-k]$$

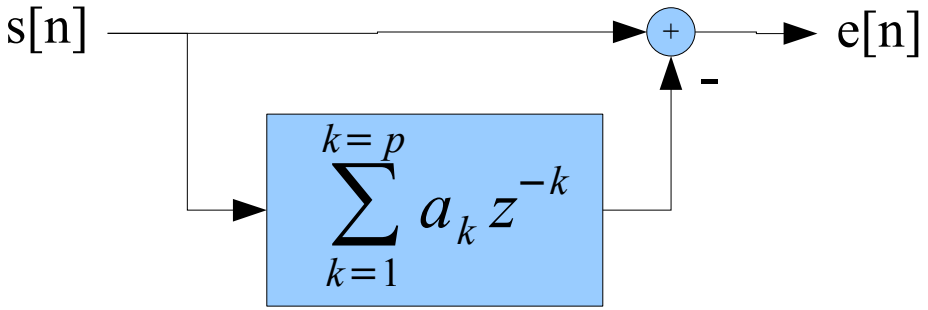
Where p is the prediction order, a_k are the prediction coefficients, and s'[n] is the predicted signal

Analysis Filter

- The prediction error, or residual, is given as:

$$e[n] = s[n] - s'[n] = s[n] - \sum_{k=1}^{k=p} a_k s[n-k]$$

Which can be realized from the following filter:

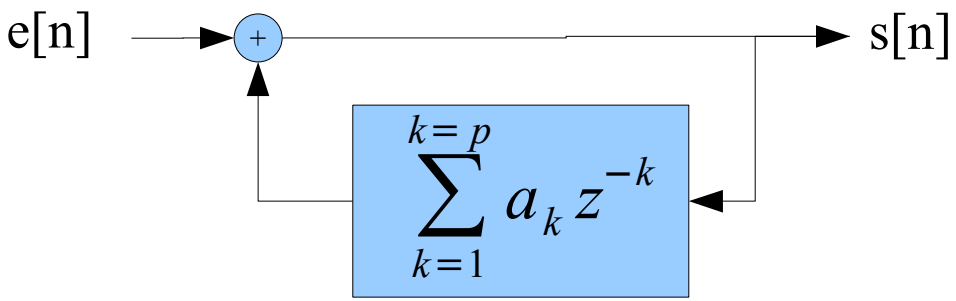


Synthesis Filter

- To synthesize the signal from the residual excitation:

$$s[n]=e[n]+s'[n]=e[n]+\sum_{k=1}^{k=p} a_k s[n-k]$$

Which can be realized from the following filter



How to Calculate Filter Coefficients?

- The LPC filter coefficients are calculated by minimizing the residual energy in the system (least squared error)

$$\sum_n e^2[n] = \sum_n \left[s[n] - \sum_{k=1}^{k=p} a_k s[n-k] \right]^2$$

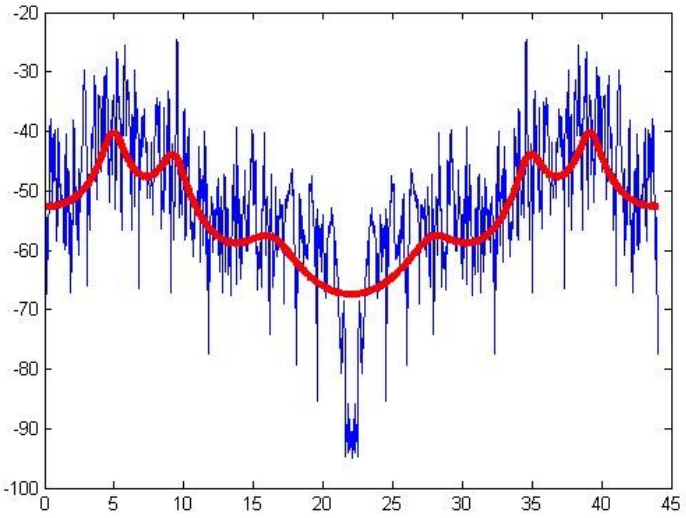
- Setting the partial derivative with respect to each a_k to zero, we get:

$$\sum_{k=1}^{k=p} a_k R[i-k] = R(i), \; i = 1, \ldots, p$$

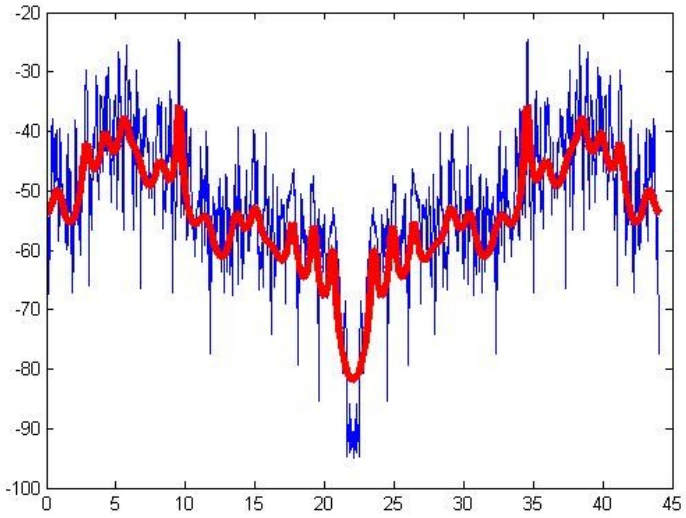
Where *R[x]* is the autocorrelation of the signal *s[n]* at lag *x*

- Here we have *p* equations in *p* unknowns, and thus we can solve for the filter coefficients a_k (these are known as autoregressive coefficients)
- The filter coefficients can be solved for efficiently using the Levinson-Durbin algorithm []

Examples



- Spectral envelope for 8th order linear predictor



- Spectral envelope for 32nd order linear predictor

Residual Coding

- The error signal is coded using "Rice" coding, a form of entropy encoding (like huffman coding)
- The first step in generating a Rice code is to set an integer parameter *M = log2(ln(2)*E(|e[n]|))*
- The residual is then divided by *M*, giving a quotient *q* and a remainder *r*
- The codeword is generated by placing *q* in unary format followed by *r* in binary format

Example: Find the rice code for *N = 17* given *M = 8*:

q = floor(N/M) = 2 = 110 (unary code)

r = N%M = 1 = 1 (modified binary code)

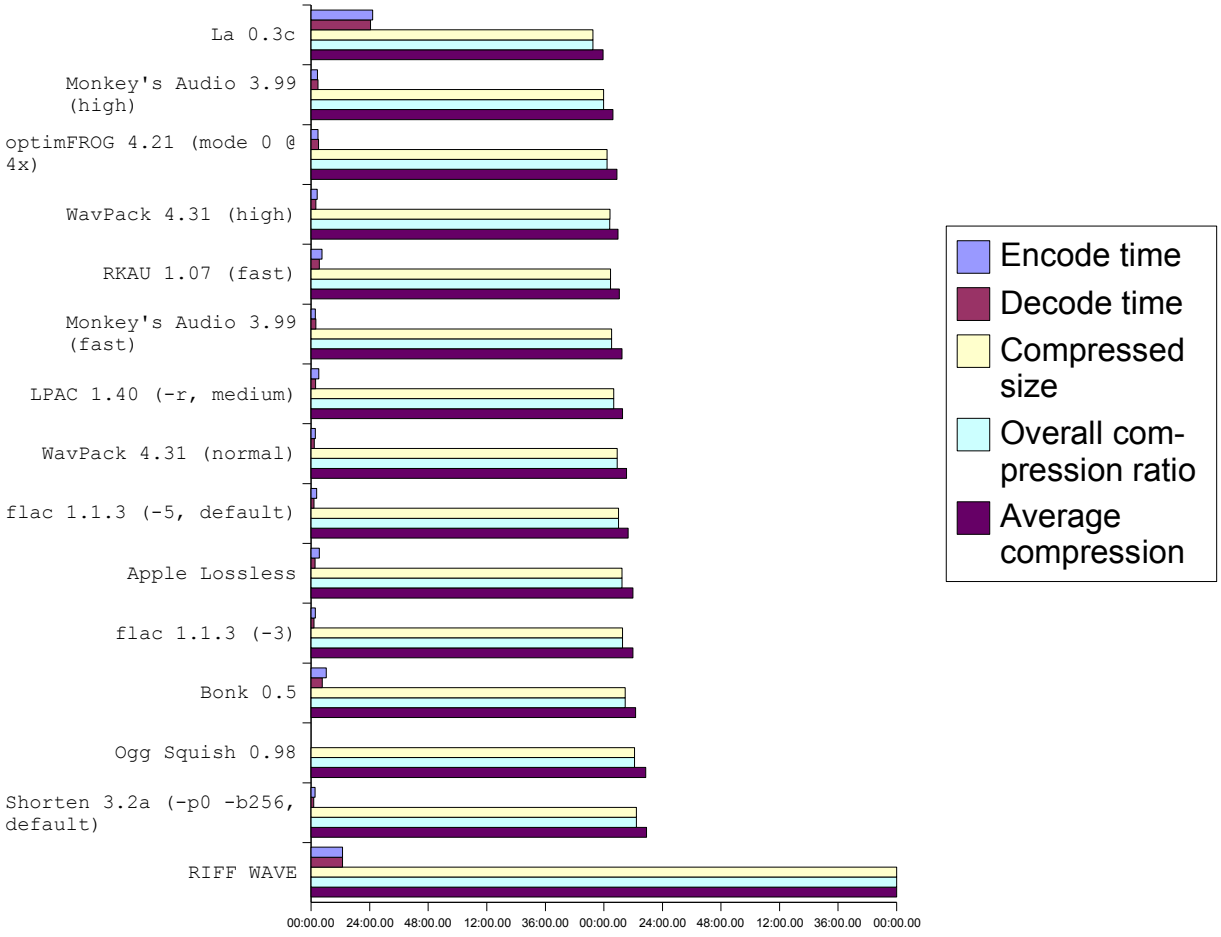
So the rice code would be *1101*

Part III: comparison to other lossless audio standards

"As far as I know, only two of the lossless encoders out there (FLAC and WavPack) are both actively developed and truly free (source code for Shorten and Monkey's Audio is available but the licenses are more restrictive). Most others give out free binaries, but without access to the source or the format specification, you are leaving your data to the whim of the maintainer for eternity; you have no alternative software to use, and no way to port the program to another OS or fix it if it breaks." - Josh Coalson

Codec	Source Available?	Player Support?	Hardware Support?	Can Stream?	Can Seek?	License Cost	OS support
flac v1.1.3	YES (OSI approved license)	YES (XMMS, AlsaPlayer, Y! Music Engine, Winamp, MacAmp Lite, dBpowerAMP, Foobar2000, QCD, Apollo, many more)	YES (Squeezebox, Sonos, PhatBox, Kenwood MusicKeg, iAudio, dozens more)	YES	YES	NONE	Linux, Windows, Mac OS X, *BSD, Solaris, OS/2, BeOS, others
WavPack v4.31	YES (OSI approved license)	YES (Winamp, foobar2000, dBpowerAMP, more)	YES (some portables via Rockbox firmware replacement)	YES	YES	NONE	Linux, Windows, Mac OS X
Shorten v3.2	YES (non-OSI license)	YES (Winamp, XMMS)	YES (some portables via Rockbox firmware replacement)	no	YES (v3 only)	non-commercial only	Linux, Windows, Mac OS 9, Mac OS X, *BSD, Solaris, others
Monkey's Audio v3.99	YES (non-OSI license)	YES (Winamp, MediaJukebox, dBpowerAMP, more)	no	no	YES	?	Windows, Linux console source
Apple Lossless	no	YES (iTunes)	YES (iPod only)	YES	YES	unavailable	Windows, Mac OS X
Ogg Squish 0.98	YES (OSI approved license)	no (?)	no	YES	YES	NONE	Linux, Windows, other UNIX
Bonk 0.5	YES (OSI approved license)	YES (XMMS)	no	no	no	?	Linux, Windows, other UNIX
La 0.3c	no	YES (Winamp, XMMS)	no	no	YES	?	Windows, Linux
optimFROG 4.21	no	YES (Winamp, XMMS)	no	no	YES	?	Windows, Linux
LPAC v1.31 (codec 3.0)	no	YES (Winamp only)	no	no?	YES	?	Windows, Linux console, Solaris console
RKAU v1.07	no	YES (Winamp only)	no	no	YES	?	Windows

Lossless Audio Codec Comparison



References

[1] Coalson, Josh. *FLAC - Free Lossless Audio Codec* 2006 [cited. Available from http://flac.sourceforge.net.

[2] Makhoul, J. 1975. Linear prediction: A tutorial review. *Proceedings of the IEEE* 63 (4):561-80

[3] Zoelzer, U. 2002. *DAFx – Digital Audio Effects*. 303-10. New York: John Wiley and Sons

[4] Golomb, S. W. 1966. Run-length encodings. *IEEE Transactions On Information Theory* 12:399-401.