

RGL Codec

(G.711 Lossless Codec)

http://www.winlab.rutgers.edu/~ramalho/rgl_codec_p19.txt
(whitepaper & code at: www.vovida.org)

Michael Ramalho
mramalho@cisco.com

RGL Codec - Use & Design Goals

Utility/Use:

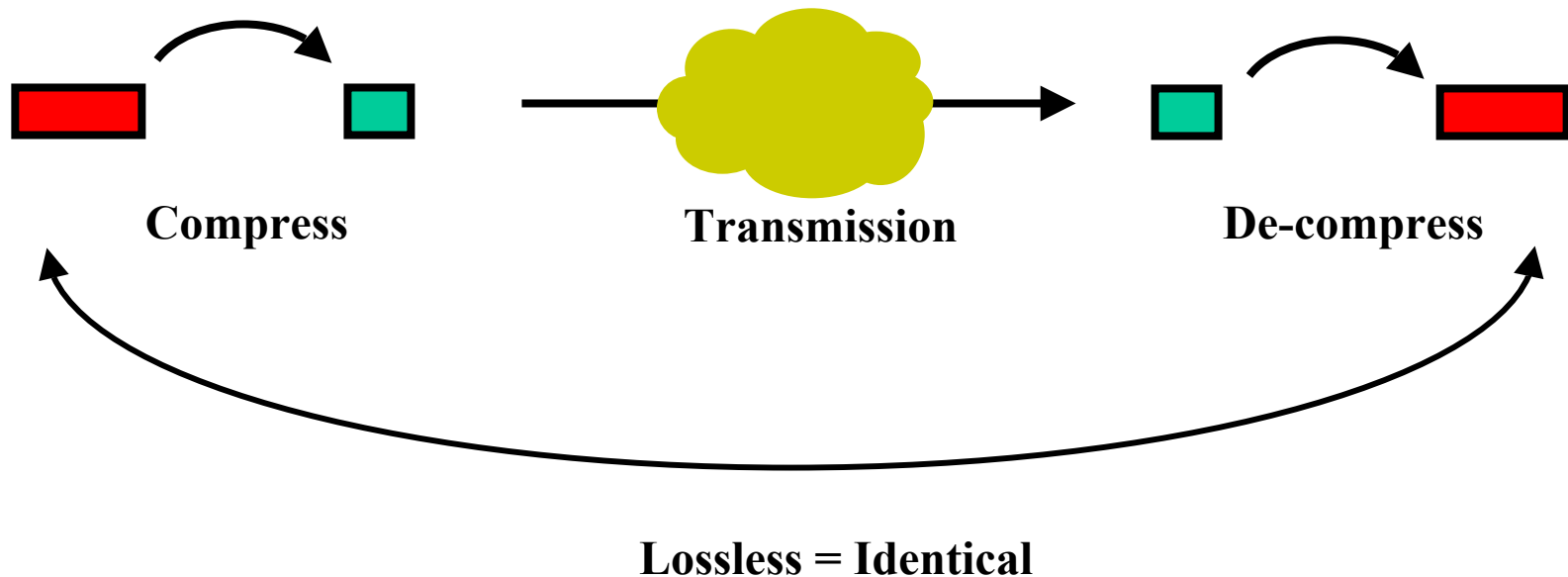
- End-to-end G.711 mandatory (e.g., V.90 modem pass-through in low-loss QoS environments).
- Applications where “instantaneous” real-time QoS bandwidth saved via compression is available for other elastic traffic.

Design Goals:

- G.711 Lossless Compression (both A-law and μ -law).
- Low complexity (on par with SRTP & DTMF decoders).
- If G.711 packet payload is “uncompressible”, expansion is limited to one byte.
- Accommodate ANY G.711 payload (assumes zero-mean acoustical input sources – but can handle any payload)
- Compression of arbitrary length G.711 samples/frame (10 msec, 20 msec, ATM:AAL2/AAL5 sizes, etc.).

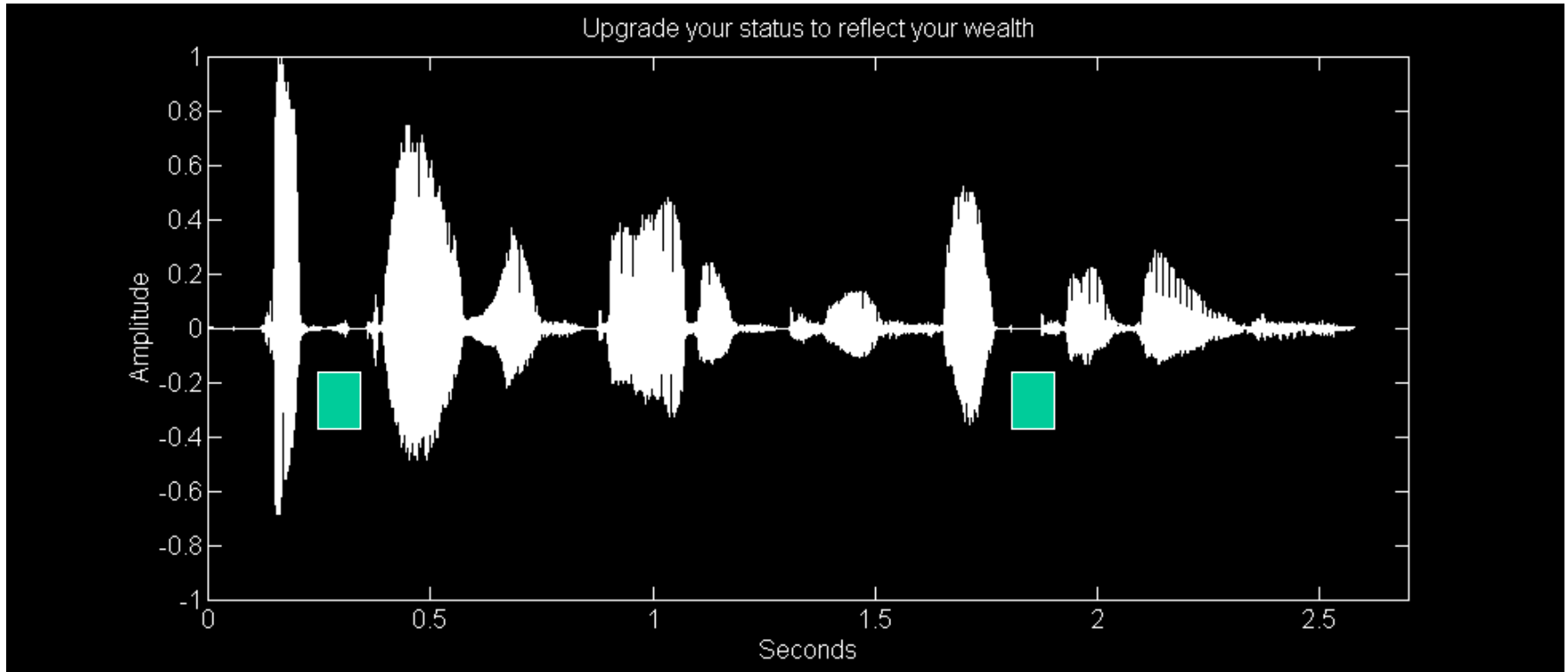
RGL Codec – Basic Idea

G.711 Lossless Codec



A “ZIP-like compression” for G.711 payloads

RGL Codec - Methodology

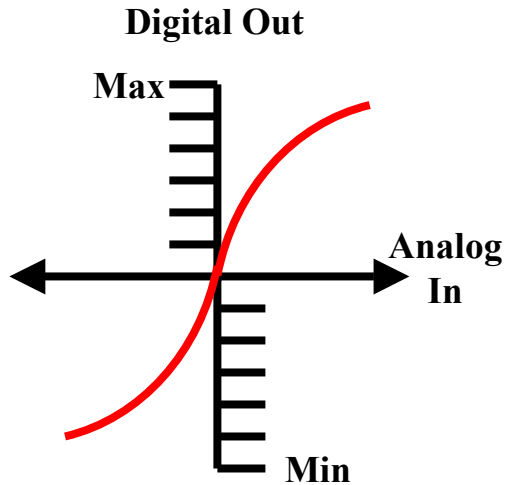


 = An approximate 10 msec segment

There are many 10 millisecond frames that exercise less than half the full input range – can save at least one bit for these segments!

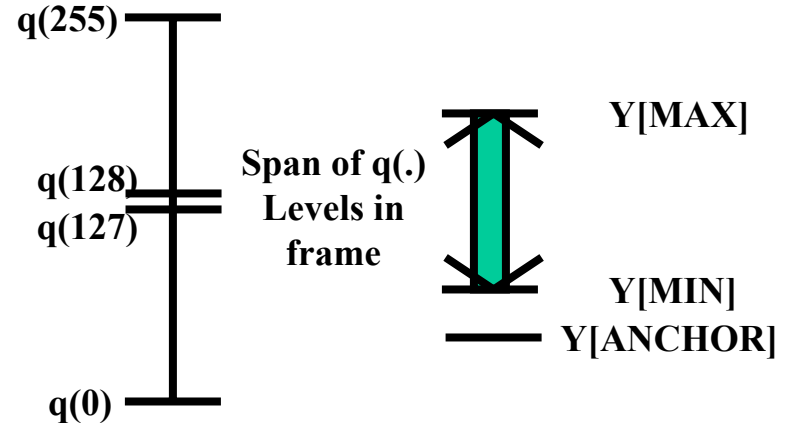
(It's a brain-dead simple concept – no rocket science needed)

RGL Codec - Methodology



G.711 is an 8-bit companding codec (approximates 13-bit linear for low signal levels)

Map to linear $q(\cdot)$



$$\text{Bits_min} = \text{ceil} (\log_2 (Y[\text{MAX}] - Y[\text{MIN}] + 1)).$$

There are 31 defined fixed anchor points defined.

Code each sample point as a count up from $Y[\text{ANCHOR}]$.

Choose $Y[\text{ANCHOR}]$ closest, but not more positive than $Y[\text{MIN}]$.

$$\text{Bits_Y}[\text{ANCHOR}] = \text{ceil} (\log_2 (Y[\text{MAX}] - Y[\text{ANCHOR}] + 1)).$$

If $\text{Bits_Y}[\text{ANCHOR}] > \text{Bits_min}$, count from $Y[\text{MIN}]$ and send $Y[\text{MIN}]$ explicitly.

All $\text{Bits_min} = 8$ encodings are anchored at $q(0)$ by default.

RGL Frame Format

First overhead byte

(always present):

{N3,N2,N1,A5,A4,A3,A2,A1}

N bits define number of bits per sample (0 – 8 bits per sample).

A bits define the anchor locations (31 anchor locations, the all ones codepoint saved to signal explicit anchor byte, if needed).

Explicit anchor byte

(only sent if Y[MIN] sent explicitly, iff all A bits =1):

{E8,E7,E6,E5,E4,E3,E2,E1}

E bits define Y[MAX] as a binary count from q(0).

Coded sampled data

(only if B != 0. Let i = sample index, M = samples in frame):

{Z[B(j)](i) ... Z1(i+M-1)}

A 3 bit per sample example:

{Z3(i) Z2(i) Z1(i) Z3(i+1) Z2(i+1) Z1(i+1) Z3(i+2) Z2(i+2) Z1(i+2) ...
Z3(i+M-1) Z2(i+M-1) Z1(i+M-1) ... (possible zero padding to align)}.

RGL Frame Format

Some Observations for RGL decoder:

- Number of bits per sample determined by N bits
- Anchor determined by A bits (or Explicit Anchor byte if all A bits = 1).
- RGL frame can be from 1 to (M+1) bytes long. For all B = 8 encodings, the (M+1) byte RGL frame has a deterministic first byte (it is always {00011110}). The RTP payload format can exploit this fact to obtain byte parity with G.711 (frames).
- Cannot determine number of samples in the received RGL payload frame (i.e., not sent as part of the RGL frame payload). This must be determined via SDP (“ptime” or the default “ptime” for codec) or sent explicitly in the corresponding RTP payload format (next draft to be discussed).
- The RGL decoder can successfully decode knowing only the RGL payload and the number of samples in the frame (static).

RGL Compression Results

Talker Loudness	Background Noise Condition	Voice Activity Factor			
		35%	40%	45%	50%
Loud	Artificial Zero	68.4%	64.0%	59.7%	55.4%
	Near Zero	44.4%	41.9%	39.4%	36.9%
	Very Low	36.3%	34.4%	32.5%	30.7%
	Low	28.2%	26.9%	25.7%	24.4%
	Moderate	19.4%	19.4%	18.8%	18.2%
Nominal	Artificial Zero	72.8%	69.0%	65.3%	61.6%
	Near Zero	48.8%	46.9%	45.0%	43.2%
	Very Low	40.7%	39.4%	38.2%	36.9%
	Low	32.5%	31.9%	31.3%	30.7%
	Moderate	24.4%	24.4%	24.4%	24.4%

- **RGL codec has high compression during non-speech.**
- **Most compression gains occur during non-speech and are highly dependent upon background noise condition.**

RGL Codec - Summary

- G.711 Lossless Compression (both A-law and μ -law). Identical fidelity to G.711 – no coding artifacts.
- Low complexity (0.16 MIPS encode, 0.14 MIPS decode).
- If G.711 packet payload is “uncompressible”, expansion is limited to one deterministic overhead byte (`{00011110}`).
- Accommodates ANY G.711 payload (assumes zero-mean acoustical input sources – but lossless under any payload).
- Compression of arbitrary length G.711 samples/frames.
- VAD not recommended, as RGL has high compression during periods of non-speech (no silence suppression induced artifacts).
- Attractive for applications where transport G.711 is mandatory and “instantaneous” real-time QoS bandwidth saved via compression can be profitably used by applications using elastic transport protocols.

For source & documentation: www.vovida.org -> RGL

RTP Payload Format for RGL Codec

http://www.winlab.rutgers.edu/~ramalho/rgl_rtp_p15.txt

Michael Ramalho
mramalho@cisco.com

RTP Payload Format for RGL Codec

Design Goals:

- Efficient coding for one RGL frame per packet.
- Optional encoding one RGL frame per packet for “8 bit per sample” that produces byte parity w.r.t. G.711 payloads of identical number of samples.
- Coding for a common “two RGL frames” per packet case (each with the identical number of samples per RGL frame).
- A “storage mode” whereby a multiplicity of RGL frames, each with an arbitrary number of samples per frame, could be placed in one RTP packet.

RTP Payload Format for RGL Codec

One RGL payload per RTP packet Cases:

Case One: X=0, M=0

Exactly one RGL frame in RTP payload. The number of samples in the RGL frame is $p_{time} * 8$ (or RGL default). No RTP header extension used. The entire RTP payload is passed to decoder at far-end.

Case Two: X=0, M=1 (*OPTIONAL*)

Exactly one "eight bit encoded" RGL frame in payload without first overhead byte (of {00011110}). The number of samples in the RGL frame is $p_{time} * 8$ (or RGL default). No RTP header extension used. The overhead byte of {00011110} is added to entire RTP payload before it is passed to the decoder at the far-end.

The M bit is thus used to obtain byte parity relative to native G.711 frames.

RTP Payload Format for RGL Codec

(The controversial, probably-not-OK, part)

Multiple RGL payload per RTP packet Cases:

Case Three: X=1, M=0

**Two RGL frames of identical number of samples in RTP payload.
Packetization details/TOC in 4-octet RTP Header Extension.**

Case Four: X=1, M=1

Arbitrary number of RGL frames in RTP payload, each of arbitrary number of samples. Packetization details/TOC in custom designed > 4 octet RTP header extension.

If this is unacceptable or undesirable, there is one way to make the RGL payload self describing.

I can use one of the 200 or so “impossible for the RGL codec” to produce “first two-octet combinations” to signal a TOC at beginning of the RTP payload.

RGL Codec

(G.711 Lossless Codec)

http://www.winlab.rutgers.edu/~ramalho/rgl_codec_p19.txt
(whitepaper & code at: www.vovida.org)

Michael Ramalho
mramalho@cisco.com