

I n t e r n a t i o n a l T e l e c o m m u n i c a t i o n U n i o n

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

G.711.0

(09/2009)

SERIES G: TRANSMISSION SYSTEMS AND MEDIA,
DIGITAL SYSTEMS AND NETWORKS

Digital terminal equipments – Coding of voice and audio
signals

Lossless compression of G.711 pulse code modulation

Recommendation ITU-T G.711.0



International
Telecommunication
Union

Documents
Document Sharing Hub

ITU-T G-SERIES RECOMMENDATIONS
TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100–G.199
GENERAL CHARACTERISTICS COMMON TO ALL ANALOGUE CARRIER-TRANSMISSION SYSTEMS	G.200–G.299
INDIVIDUAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON METALLIC LINES	G.300–G.399
GENERAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON RADIO-RELAY OR SATELLITE LINKS AND INTERCONNECTION WITH METALLIC LINES	G.400–G.449
COORDINATION OF RADIOTELEPHONY AND LINE TELEPHONY	G.450–G.499
TRANSMISSION MEDIA AND OPTICAL SYSTEMS CHARACTERISTICS	G.600–G.699
DIGITAL TERMINAL EQUIPMENTS	G.700–G.799
General	G.700–G.709
Coding of voice and audio signals	G.710–G.729
Principal characteristics of primary multiplex equipment	G.730–G.739
Principal characteristics of second order multiplex equipment	G.740–G.749
Principal characteristics of higher order multiplex equipment	G.750–G.759
Principal characteristics of transcoder and digital multiplication equipment	G.760–G.769
Operations, administration and maintenance features of transmission equipment	G.770–G.779
Principal characteristics of multiplexing equipment for the synchronous digital hierarchy	G.780–G.789
Other terminal equipment	G.790–G.799
DIGITAL NETWORKS	G.800–G.899
DIGITAL SECTIONS AND DIGITAL LINE SYSTEM	G.900–G.999
MULTIMEDIA QUALITY OF SERVICE AND PERFORMANCE – GENERIC AND USER-RELATED ASPECTS	G.1000–G.1999
TRANSMISSION MEDIA CHARACTERISTICS	G.6000–G.6999
DATA OVER TRANSPORT – GENERIC ASPECTS	G.7000–G.7999
PACKET OVER TRANSPORT ASPECTS	G.8000–G.8999
ACCESS NETWORKS	G.9000–G.9999

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T G.711.0

Lossless compression of G.711 pulse code modulation

Summary

Recommendation ITU-T G.711.0 describes a lossless compression scheme of a G.711 bitstream, mainly aimed for transmission over IP (e.g., VoIP).

The coder operates on frame lengths of 40, 80, 160, 240 and 320 samples, has a maximum algorithmic delay equal to the frame length, and has a worst-case computational complexity of less than 1.7 weighted million operations per second (WMOPS) for encoder plus decoder.

This Recommendation includes an electronic attachment containing a non-exhaustive set of test signals for use with the ANSI C code. ANSI C source code is provided for the fixed-point arithmetic implementation of the specification.

Source

Recommendation ITU-T G.711.0 was approved on 22 September 2009 by ITU-T Study Group 16 (2009-2012) under Recommendation ITU-T A.8 procedures.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at <http://www.itu.int/ITU-T/ipr/>.

© ITU 2010

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

CONTENTS

Page

1	Scope	1
2	Reference	1
3	Definitions	1
4	Abbreviations and acronyms	1
5	Conventions	2
6	General description of the ITU-T G.711.0 coder	4
6.1	Encoder	4
6.2	Decoder	5
6.3	Supported frame lengths	6
6.4	Bit rate	6
6.5	Algorithmic delay	6
6.6	Computational complexity and memory requirements	7
6.7	Coder description	7
6.8	Mapping functions	7
6.9	Variable-length coding schemes	10
7	Functional description of the encoder	10
7.1	Prefix codes for frame length	10
7.2	Prefix codes associated with each encoding tool	11
7.3	Encoding tool selection	12
7.4	Uncompressed coding tool	14
7.5	Constant value coding tools	14
7.6	Plus-Minus zero Rice coding tool	15
7.7	Binary coding tool	16
7.8	Pulse mode coding tool	16
7.9	Value-location coding tool	18
7.10	Mapped domain LP coding tool	21
7.11	Fractional-bit coding tool	38
7.12	Min-Max level coding tool	41
7.13	Direct LP coding tool	45
8	Functional description of the decoder	45
8.1	Obtaining the frame length and the decoding tool from a prefix code	45
8.2	Uncompressed decoding tool	46
8.3	Constant value decoding tools	46
8.4	PM zero Rice decoding tool	46
8.5	Binary decoding tool	46
8.6	Pulse mode decoding tool	46
8.7	Value-location decoding tool	47

	Page
8.8 Mapped domain LP decoding tool	49
8.9 Fractional bit decoding tool.....	54
8.10 Min-Max level decoding tool	55
8.11 Direct LP decoding tool.....	55
9 Bit-exact description of the ITU-T G.711.0 codec	55
9.1 Use of the simulation software	56
9.2 Organization of the simulation software	57
Electronic attachment – Fixed-point reference implementation and associated test signals	

Copyright © 2009 ITU-T. All rights reserved.

Recommendation ITU-T G.711.0

Lossless compression for G.711 pulse code modulation

1 Scope

This Recommendation contains the description of a lossless compression scheme of an ITU-T G.711 bitstream.

This Recommendation is organized as follows. The references, definitions, abbreviations, acronyms, and conventions used throughout this Recommendation are defined in clauses 2, 3, 4, and 5, respectively. Clause 6 gives a general outline of the ITU-T G.711.0 algorithm. The ITU-T G.711.0 encoder and decoder principles are discussed in clauses 7 and 8, respectively. Clause 9 describes the software that defines this coder in 16-32 bit fixed-point arithmetic.

This Recommendation contains an electronic attachment with a fixed-point reference implementation and a non-exhaustive set of test signals for use with the reference implementation.

2 Reference

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

- [ITU-T G.191] Recommendation ITU-T G.191 (2005), *Software tools for speech and audio coding standardization*.
- [ITU-T G.711] Recommendation ITU-T G.711 (1988), *Pulse code modulation (PCM) of voice frequencies*.

3 Definitions

This Recommendation does not introduce new definitions.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

IP	Internet Protocol
LLC	LossLess Compression
LP	Linear Prediction
LPC	Linear Predictive Coding
LSB	Least Significant Bit
LTP	Long-Term Prediction
MSB	Most Significant Bit
PARCOR	PARtial autoCORrelation
PCM	Pulse Code Modulation
VoIP	Voice over IP

00000000000000000000000000000000

5 Conventions

The notational conventions are detailed below.

- Time domain signals are denoted by their symbol and a sample index between parentheses, e.g., $s(n)$. The variable n is used as a sample index.
- The symbol $\hat{\cdot}$ identifies a quantized version of a parameter (e.g., $\hat{g}_c$) or a predicted sample value (e.g., $\hat{x}(n)$).
- Parameter ranges are given between square brackets, and include the boundaries (e.g., $[0.6, 0.9]$). When one of the boundaries is denoted by using a parenthesis, the boundary at that end is not included in the range (e.g., $[0.6, 0.9)$).
- The sign function gives the polarity of the value and is denoted as $\text{sgn}(x)$, where

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases}$$
- Operator $\max\{x(n) | n=0, \dots, N-1\}$ denotes the maximum value of $x(n)$, $0 \leq n \leq N-1$.
- Operator $\min\{x(n) | n=0, \dots, N-1\}$ denotes the minimum value of $x(n)$, $0 \leq n \leq N-1$.
- Integer operator $\lceil x \rceil$ denotes a ceiling function, rounding x towards plus infinity ($\lceil x \rceil = \min\{n \in \{\dots, -2, -1, 0, 1, 2, \dots\} | x \leq n\}$).
- Integer operator $\lfloor x \rfloor$ denotes a floor function, rounding x towards minus infinity ($\lfloor x \rfloor = \max\{n \in \{\dots, -2, -1, 0, 1, 2, \dots\} | x \geq n\}$).
- In some parts, bit operators are used, where $\otimes$ and $\oplus$ represent the AND bit-operator and the XOR bit-operator, respectively.
- The constants with "0x" prefix mean that the values are expressed in hexadecimal.
- N -bit right-shift operations of a variable x are denoted as multiplications with floor of 2 to the power of $-N$, i.e., $\lfloor 2^{-N} x \rfloor$.
- In the 16-bit fixed-point ANSI C implementation, the floating-point numbers are rounded to respective 16/32-bit representations.

Table 5-1 lists the most relevant symbols used throughout this Recommendation.

In this Recommendation, "plus zero" and "minus zero" (noted 0^+ and 0^- , respectively) correspond to the ITU-T G.711 encoded value for the positive/negative input value that will be decoded by the smallest magnitude value at the ITU-T G.711 decoder.

Table 5-1 – Glossary of most relevant symbols

Type	Name	Description
Signals	$I_A(n)$	Input ITU-T G.711 A-law samples to be encoded
	$I_\mu(n)$	Input ITU-T G.711 μ -law samples to be encoded
	$x_A(n)$	ITU-T G.711 A-law samples
	$x_\mu(n)$	ITU-T G.711 μ -law samples
	$x_{\text{int8}}(n)$	Samples in <i>int8</i> domain ($-128 \leq x_{\text{int8}}(n) \leq 127$)
	$\hat{x}_{\text{int8}}(n)$	Predicted samples in <i>int8</i> domain
	$x_{\text{PCM}}(n)$	Samples in uniform (linear) PCM domain
	$\hat{x}_{\text{PCM}}(n)$	Predicted samples in uniform (linear) PCM domain
	$\tilde{x}_{\text{PCM}}(n)$	Windowed samples of $x_{\text{PCM}}(n)$
	$\bar{x}_{\text{PCM}}(n)$	Down-sampled PCM signal
	$\bar{x}_{\text{LTP}}(n)$	LTP contribution
	$\hat{x}_{\text{LTP}}(n)$	LTP predicted PCM signal
	$r(n)$	Linear prediction residual samples in <i>int8</i> domain
	$r_{\text{PCM}}(n)$	Short-term prediction residual samples in uniform PCM domain
	$r_{\text{LTP}}(n)$	Long-term prediction residual samples in <i>int8</i> domain
Coefficients	$c(i)$	Autocorrelation coefficients of the windowed signal $\tilde{x}_{\text{PCM}}(n)$
	$c'(i)$	Bandwidth expanded autocorrelation coefficients
	k_i	PARCOR coefficients
	$\hat{k}_i$	Quantized PARCOR coefficients
	$a_i^{[j]}$	Linear prediction coefficients for <i>j</i> -th prediction
Parameters	n	Index of sample
	N	Number of samples in a frame
	R	Data range
	$X_{\min}$	The minimum sample value of $x_{\text{int8}}(n)$
	$X_{\max}$	The maximum sample value of $x_{\text{int8}}(n)$
	X_{ANCHOR}	An anchor value
	P	Prediction order
	$\hat{P}$	Quantized prediction order
	$P_{\max}$	Maximum possible prediction order
	$p(j)$	Candidate values of prediction order
	N_{sfr}	Number of sub-frames in LTP analysis
	N'_{sfr}	Number of sub-frames in prediction residual coding

Table 5-1 – Glossary of most relevant symbols

Type	Name	Description
Special values	0^+	Denotes plus-zero value ("zero" decoder output for positive PCM input value: 0xd5 for A-law and 0xff for μ -law)
	0^-	Denotes minus-zero value ("zero" G.711 decoder output for negative PCM input value: 0x55 for A-law and 0x7f for μ -law)
	0_m	More zero: The value of 0^+ or 0^- which has more occurrences in the frame
	0_l	Less zero: The value of 0^+ or 0^- which has less occurrences in the frame
Other	N/A	Not applicable

6 General description of the ITU-T G.711.0 coder

The ITU-T G.711 lossless codec of this Recommendation is implemented in fixed-point arithmetic using version 2.2 of the basic operators defined in the [ITU-T G.191] software tool library. This Recommendation provides the detailed algorithm description.

The lossless compression techniques employed in this Recommendation are most effective when the ITU-T G.711 symbols are a digital representation of a zero-mean acoustic source such as speech or audio. When this is the case, overall compression is expected. However, the ITU-T G.711.0 codec has been designed to encode any supported number of ITU-T G.711 symbols independent of this zero-mean assumption. A limited number of encoding techniques have been employed in a desire to limit the implementation complexity.

The ITU-T G.711.0 codec has also been designed to be stateless – the encoding of an input frame of ITU-T G.711 symbols is independent of any previous/look-ahead ITU-T G.711 input frame. This stateless property ensures that there will be no "error propagation". Better compression ratio may have been obtained when using past information but the design constraint of stateless lossless encoding makes this Recommendation very suitable for packet-based applications.

6.1 Encoder

Figure 6-1 shows the high-level block diagram of the encoder.

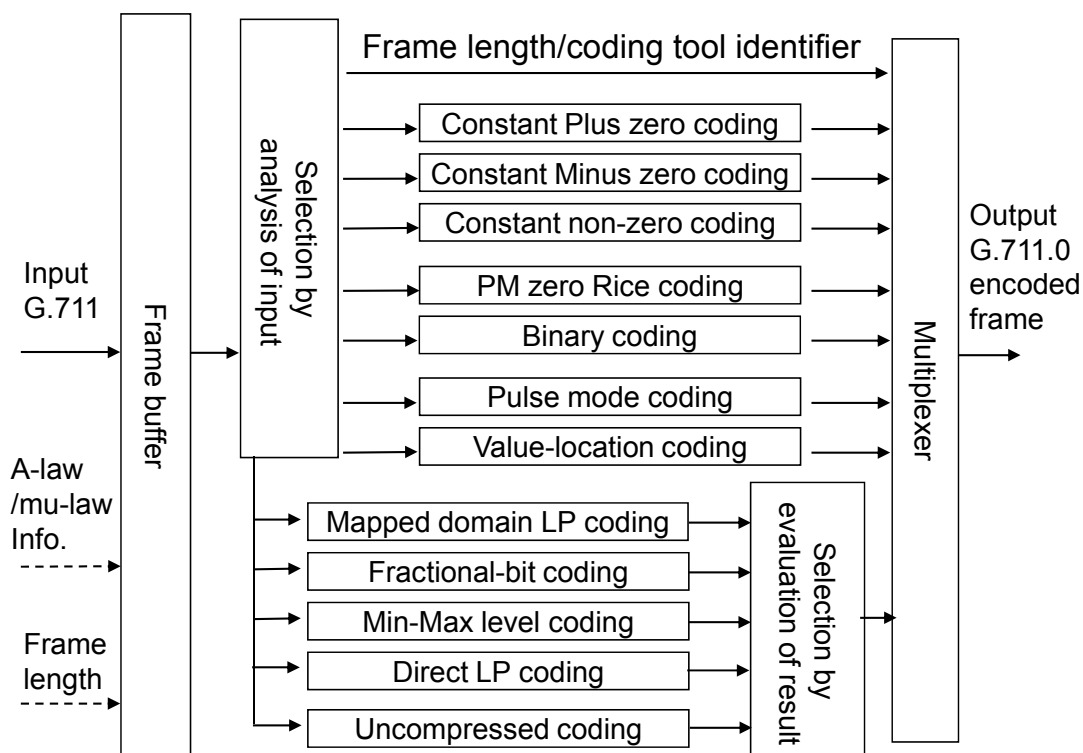


Figure 6-1 – High-level encoder block diagram

This Recommendation selects one of the coding methods shown in Figure 6-1 to create the ITU-T G.711.0 encoded output frame. Prior to the selection, one or more of the coding methods may be computed or partially computed. The decision process is fully described in clause 7.3. After the method selection, a "prefix code" is placed at the top of the chosen encoding information and sent as a part of the ITU-T G.711.0 encoded output frame. This prefix code contains information that defines the frame length, the chosen encoding method, and other side information designated to the chosen coding method. The prefix code definition is in clause 7.1 for the frame length and in clause 7.2 for the encoding methods.

In the remainder of this Recommendation, the word "tool" will be used to denote the computational or algorithmic machinery necessary to encode or decode using a particular encoding method. For example, a "Pulse mode encoding" is accomplished by a "Pulse mode encoding tool".

6.2 Decoder

Figure 6-2 shows the high-level block diagram of the decoder.

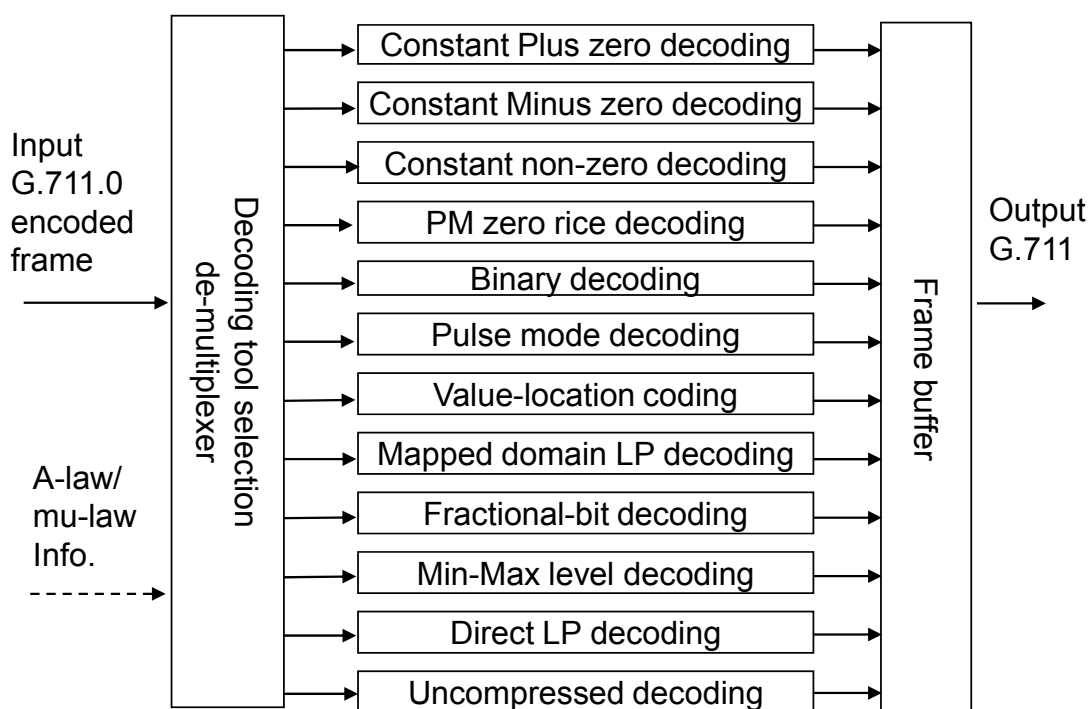


Figure 6-2 – High-level decoder block diagram

The ITU-T G.711.0 decoder reads the prefix code in the received ITU-T G.711.0 encoded frame and passes the necessary information to the appropriate decoding tool. The decoding tool reproduces the ITU-T G.711 samples using the received ITU-T G.711.0 encoded frame.

6.3 Supported frame lengths

The ITU-T G.711.0 coder supports five frame lengths: 40, 80, 160, 240, and 320 samples per frame. As there are no inter-frame dependencies, the frame length can be changed for every frame. The frame length encoding is described in clause 7.1. Thus, the decoder can decode any frames without out-of-band signalling of the frame length information. ITU-T G.711.0 encoded frames are therefore "self-describing" in terms of the number of samples they represent.

6.4 Bit rate

The ITU-T G.711.0 codec is a variable bit rate. The size of the produced ITU-T G.711.0 bitstream depends on the input signal characteristics. The minimum size of an encoded frame is one octet (8 bits). The maximum size of an encoded frame is the input frame data size plus one octet and occurs when the input frame is uncompressible by any of the encoding tools. The size of an encoded frame is expressed in octets.

6.5 Algorithmic delay

The ITU-T G.711.0 coder has algorithmic delays of 5, 10, 20, 30, and 40 ms which correspond to frame lengths of 40, 80, 160, 240 and 320 samples at 8000 Hz, respectively. There is no additional algorithmic delay contribution other than this frame buffering.

6.6 Computational complexity and memory requirements

The observed worst-case complexity of the ITU-T G.711.0 coder (encoder plus decoder) is 1.667 WMOPS based on the basic operators of ITU-T software tool library STL2005 v2.2 in [ITU-T G.191]. The worst computational complexity is detailed in Table 6-1 for either μ -law or A-law. The ITU-T G.711.0 memory requirements in octets are given in Table 6-2. Note that the RAM figures are based on most of all variables including array and single variables. Here, 'k' represents 1000 units.

Table 6-1 – Worst computational complexity of ITU-T G.711.0 coder (WMOPS)

Encoder	Decoder
1.0785	0.5887

Table 6-2 – Memory requirements of ITU-T G.711.0 coder

	Encoder	Decoder
Static RAM (k octets)	0.01	0
Scratch RAM (k octets)	3.59	1.37
Data ROM (k octets)	5.48	
Program ROM (number of basic ops)	3554	

6.7 Coder description

The description of the coding algorithm of this Recommendation is made in terms of bit-exact fixed-point mathematical operations. The ANSI C code indicated in clause 9, which constitutes an integral part of this Recommendation, reflects this bit exact, fixed-point descriptive approach. The mathematical descriptions of the encoder and decoder can be implemented in other fashions, possibly leading to a codec implementation not complying with this Recommendation. Therefore, the algorithm description of the ANSI code of clause 9 shall take precedence over the mathematical descriptions whenever discrepancies are found. A non-exhaustive set of test signals, which can be used with the ANSI C code, is available as an electronic attachment to this Recommendation.

6.8 Mapping functions

This clause describes functions that map an ITU-T G.711 sample to the various formats required in the different coding tools of the ITU-T G.711.0 encoder and decoder.

6.8.1 Conversion from A-law/ μ -law to uniform PCM

6.8.1.1 Conversion from A-law to uniform PCM

A conversion function $f_{A \rightarrow \text{PCM}}$ which maps an A-law sample, $x_A(n)$, to a uniform PCM sample, $x_{\text{PCM}}(n)$, is defined as follows:

$$x_{\text{PCM}}(n) = f_{A \rightarrow \text{PCM}}(x_A(n)) \quad (6-1)$$

where, given $s_A = x_A(n) \otimes 0x80$ and $y_A = (x_A(n) \oplus 0x55) \otimes 0x7F$,

$$e_A = \left\lfloor \frac{y_A}{2^4} \right\rfloor$$

$$m_A = y_A \otimes 0x0F$$

$$\text{if } e_A > 0 \quad (6-2)$$

$$x_{\text{PCM}}(n) = \begin{cases} \left\{ 2^{e_A-1} \cdot (2^4 m_A + 8 + 256) \right\} / 8 & \text{if } s_A = 0 \times 80 \\ - \left\{ 2^{e_A-1} \cdot (2^4 m_A + 8 + 256) \right\} / 8 & \text{if } s_A = 0 \end{cases}$$

else

$$x_{\text{PCM}}(n) = \begin{cases} \left\{ 2^4 m_A + 8 \right\} / 8 & \text{if } s_A = 0 \times 80 \\ - \left\{ 2^4 m_A + 8 \right\} / 8 & \text{if } s_A = 0 \end{cases}$$

Here, s_A , e_A and m_A are the sign, the exponent and the mantissa of $x_A(n)$, respectively.

6.8.1.2 Conversion from μ -law to uniform PCM

A conversion function $f_{\mu \rightarrow \text{PCM}}$ which maps a μ -law sample, $x_\mu(n)$, to a uniform PCM sample, $x_{\text{PCM}}(n)$, is defined as follows:

$$x_{\text{PCM}}(n) = f_{\mu \rightarrow \text{PCM}}(x_\mu(n)) \quad (6-3)$$

where, given $s_\mu = x_\mu(n) \otimes 0 \times 80$ and $y_\mu = (x_\mu(n) \oplus 0 \times 7F) \otimes 0 \times 7F$,

$$\begin{aligned} e_\mu &= \left\lfloor \frac{y_\mu}{2^4} \right\rfloor \\ m_\mu &= y_\mu \otimes 0 \times 0F \\ x_{\text{PCM}}(n) &= \begin{cases} \left\{ 2^e \cdot (2^3 m_\mu + 128 + 4) - 132 \right\} / 4 & \text{if } s_\mu = 0 \times 80 \\ - \left\{ 2^e \cdot (2^3 m_\mu + 128 + 4) - 132 \right\} / 4 & \text{if } s_\mu = 0 \end{cases} \end{aligned} \quad (6-4)$$

Here, s_μ , e_μ and m_μ are the sign, the exponent and the mantissa of $x_\mu(n)$, respectively.

6.8.2 Conversion from uniform PCM to A-law/ μ -law

6.8.2.1 Conversion from uniform PCM to A-law

A conversion function $f_{\text{PCM} \rightarrow A}$ which maps a uniform PCM sample, $x_{\text{PCM}}(n)$, to an A-law sample, $x_A(n)$, is defined as follows:

$$x_A(n) = f_{\text{PCM} \rightarrow A}(x_{\text{PCM}}(n)) \quad (6-5)$$

where, given $m_A = \begin{cases} \min\{x_{\text{PCM}}(n), 4095\} & \text{if } x_{\text{PCM}}(n) \geq 0 \\ -\max\{x_{\text{PCM}}(n), -4096\} - 1 & \text{if } x_{\text{PCM}}(n) < 0 \end{cases}$, $e_A = \left\lceil \log_2 \left(\frac{m_A}{2^6} + 1 \right) \right\rceil$, and

$$\begin{aligned} y_A &= \left(2^4 e_A + \left\lfloor \frac{m_A}{2^{e_A+1}} \right\rfloor \right) \oplus 0 \times 55, \\ x_A(n) &= \begin{cases} y_A \oplus 0 \times 80 & \text{if } x_{\text{PCM}}(n) \geq 0 \\ y_A & \text{if } x_{\text{PCM}}(n) < 0 \end{cases} \end{aligned} \quad (6-6)$$

6.8.2.2 Conversion from uniform PCM to μ -law

A conversion function $f_{\text{PCM} \rightarrow \mu}$ which maps a uniform PCM sample, $x_{\text{PCM}}(n)$, to a μ -law sample, $x_\mu(n)$, is defined as follows:

$$x_\mu(n) = f_{\text{PCM} \rightarrow \mu}(x_{\text{PCM}}(n)) \quad (6-7)$$

where given $m_\mu = \begin{cases} \min\{x_{\text{PCM}}(n) + 33, 8191\} & \text{if } x_{\text{PCM}}(n) \geq 0 \\ -\max\{x_{\text{PCM}}(n) - 33, -8192\} - 1 & \text{if } x_{\text{PCM}}(n) < 0 \end{cases}$, $e_\mu = \left\lceil \log_2 \left(\frac{m_\mu}{2^6} + 1 \right) \right\rceil$, and

$$y_{\mu} = \left(2^4(e_{\mu} - 1) + \left\lfloor \frac{m_{\mu}}{2^{e_{\mu}+1}} \right\rfloor \right) \oplus 0x7F,$$

$$x_{\mu}(n) = \begin{cases} y_{\mu} \oplus 0x80 & \text{if } x_{PCM}(n) \geq 0 \\ y_{\mu} & \text{if } x_{PCM}(n) < 0 \end{cases} \quad (6-8)$$

6.8.3 Conversion from A-law/ μ -law to *int8*

An *int8* frame data is made of ITU-T G.711 A-law $x_A(n)$ or μ -law $x_{\mu}(n)$ symbols converted to *signed* 8 bit values (*int8*) where the most positive ITU-T G.711 codepoint has been mapped to 127 and the most negative codepoint has been mapped to -128.

6.8.3.1 Conversion from A-law to *int8*

A conversion function $f_{A \rightarrow \text{int8}}$ which maps an A-law sample, $x_A(n)$, to an *int8* sample, $x_{\text{int8}}(n)$, is defined as follows:

$$x_{\text{int8}}(n) = f_{A \rightarrow \text{int8}}(x_A(n)), \quad (6-9)$$

where, given $s_A = x_A(n) \otimes 0x80$ and $y_A = (x_A(n) \oplus 0x55) \otimes 0x7F$,

$$x_{\text{int8}}(n) = \begin{cases} y_A & \text{if } s_A = 0 \times 80 \\ -y_A - 1 & \text{if } s_A = 0 \end{cases} \quad (6-10)$$

6.8.3.2 Conversion from μ -law to *int8*

A conversion function $f_{\mu \rightarrow \text{int8}}$ which maps a μ -law sample, $x_{\mu}(n)$, to an *int8* sample, $x_{\text{int8}}(n)$, is defined as follows:

$$x_{\text{int8}}(n) = f_{\mu \rightarrow \text{int8}}(x_{\mu}(n)) \quad (6-11)$$

where, given $s_{\mu} = x_{\mu}(n) \otimes 0x80$ and $y_{\mu} = (x_{\mu}(n) \oplus 0x7F) \otimes 0x7F$,

$$x_{\text{int8}}(n) = \begin{cases} y_{\mu} & \text{if } s_{\mu} = 0 \times 80 \\ -y_{\mu} - 1 & \text{if } s_{\mu} = 0 \end{cases} \quad (6-12)$$

6.8.4 Conversion from *int8* to A-law/ μ -law

6.8.4.1 Conversion from *int8* to A-law

A conversion function $f_{\text{int8} \rightarrow A}$ which maps an *int8* sample, $x_{\text{int8}}(n)$, to an A-law sample, $x_A(n)$, is defined as follows:

$$x_A(n) = f_{\text{int8} \rightarrow A}(x_{\text{int8}}(n)) \quad (6-13)$$

where

$$x_A(n) = f_{A \rightarrow \text{int8}}(x_{\text{int8}}(n) + 128) + 128 \quad (6-14)$$

6.8.4.2 Conversion from *int8* to μ -law

A conversion function $f_{\text{int8} \rightarrow \mu}$ which maps an *int8* sample, $x_{\text{int8}}(n)$, to a μ -law sample, $x_{\mu}(n)$, is defined as follows:

$$x_{\mu}(n) = f_{\text{int8} \rightarrow \mu}(x_{\text{int8}}(n)) \quad (6-15)$$

where

$$x_{\mu}(n) = f_{\mu \rightarrow \text{int8}}(x_{\text{int8}}(n) + 128) + 128 \quad (6-16)$$

6.9 Variable-length coding schemes

This clause describes variable-length coding schemes used in the ITU-T G.711.0 encoder and decoder.

6.9.1 Unary coding

In this Recommendation, unary code of an integer value v ($v \geq 0$) is defined as v bits of zeros followed by a single 1 in binary representation. With this unary code, the code length of the integer value v is $v+1$ bits.

To decode the unary code bitstream, the number of sequential zeros followed by a single 1 is searched and the number of sequential zeros, v , should be the decoded value.

"Unary(v)" denotes the unary encoded bitstream of an integer value v ($v \geq 0$).

6.9.2 Rice coding

Rice code, also known as Golomb-Rice code, of an integer value v ($v \geq 0$) is defined as follows when a Rice parameter S ($S \geq 0$) is given:

- 1) Separate the S LSBs of v and set them to j . j is a remainder value represented in S -bits. These bits become the LSBs of the Rice code.
- 2) Obtain a quotient $k = \left\lfloor \frac{v}{2^S} \right\rfloor$ and unary encode (see clause 6.9.1) the quotient k in $k+1$ bits. These bits become the MSB of the Rice code.

The Rice code bitstream is decoded as follows for a given Rice parameter S ($S \geq 0$):

- 1) Obtain the value of k by searching the number of sequential zeros followed by a single 1.
- 2) Read the following S bits and set them to j .
- 3) Obtain $v = k \times 2^S + j$.

"Rice(S,v)" denotes the Rice encoded bitstream of an integer value v ($v \geq 0$), with a given Rice parameter S . The first argument of the function is a Rice parameter and the second argument of the function is the value to be encoded.

7 Functional description of the encoder

7.1 Prefix codes for frame length

As described in clause 6.3, an ITU-T G.711.0 codec supports frame lengths of 40, 80, 160, 240 and 320 samples per frame. Table 7-1 shows the associated prefix code for each supported frame length. This identifier is placed at the beginning of the bitstream, i.e., the first octet of an encoded bitstream. In addition, the code 0x00 indicates that the frame length is 0 as a special case. The code 0x00 at the beginning of a potential ITU-T G.711.0 encoded frame indicates no operation of the decoding process. Therefore, the value 0x00 should be used for any padding octets required by the encoding system between/after one or more ITU-T G.711.0 encoded frames (e.g., to word-align compressed payload into the transmitting system word convention).

Table 7-1 – List of supported frame lengths and corresponding prefix codes

Frame length (<i>N</i> samples)	Prefix code (in binary representation)
0	0000 0000
40	01-- ----
80	10-- ----
160	11-- ----
240	0010 ----
320	0011 ----

7.2 Prefix codes associated with each encoding tool

Table 7-2 shows a list of encoding tools and their corresponding prefix codes. This prefix is placed immediately after the frame-length prefix code as the first octet (and continued to the second octet for some of coding tools) of the encoded bitstream. The encoded bitstream produced by the individual coding tools are placed right after the prefix codes.

Table 7-2 – List of encoding tools and corresponding tool prefix codes

Tool type		Prefix code for ITU-T G.711.0 encoded frame (initial bits, in binary representation)		Note
		<i>N</i> =40, 80, 160	<i>N</i> =240, 320	
Uncompressed coding		--00 0000	---- 0000	See clause 7.4
Constant coding tools	Constant plus zero coding	--00 0001	---- 0001	See clause 7.5
	Constant minus zero coding	--00 0010	---- 0010	
	Constant non-zero coding	--00 0011	---- 0011	
Plus Minus (PM) zero only	Binary coding	--00 0100	---- 0100	See clause 7.7
	PM zero Rice coding (Minus ≤ Plus)	--01 0ss	---- 100s s	Following two bits after the prefix code ss != 00 See clause 7.6
	PM zero Rice coding (Minus > Plus)	--01 1ss	---- 101s s	
Plus Minus zero only except one sample	Pulse mode coding (Minus < Plus)	--01 000	---- 1000 0	See clause 7.8
	Pulse mode coding (Minus > Plus)	--01 100	---- 1010 0	
Value-location coding		--00 0110	---- 0101	See clause 7.9
Mapped domain LP coding	LTP: enabled	1100 1	---- 011	Only for <i>N</i> ≥160 See clause 7.10
	LTP: disabled	--1	---- 11	See clause 7.10
Fractional bit coding		0000 0010 to 0001 1111		See clause 7.11
Min-Max level coding		0100 0101	N/A	Only for <i>N</i> =40 See clause 7.12
Direct LP coding		0100 1	N/A	Only for <i>N</i> =40 See clause 7.13
NOTE – The first bits "--" (for <i>N</i> =40, 80, 160) or "----" (for <i>N</i> =240, 320) are the prefix codes representing the frame length, given in Table 7-1.				

7.3 Encoding tool selection

Table 7-3 shows the possible tools for each frame length. A box marked with "x" indicates that the tool is available for the specific frame length. A box marked with "N/A" indicates that this tool is not used for the specific frame length.

Table 7-3 – Possible tools for each frame length

Tool type	Frame length (<i>N</i> samples)				
	40	80	160	240	320
Uncompressed coding	x	x	x	x	x
Constant coding tools	x	x	x	x	x
Plus Minus (PM) zero only	x	x	x	x	x
Plus Minus zero only except one sample	x	x	x	x	x
Value-location coding	N/A	x	x	x	x
Mapped domain LP coding	x	x	x	x	x
Fractional bit coding	x	x	x	x	x
Min-Max level coding	x	N/A	N/A	N/A	N/A
Direct LP coding	x	N/A	N/A	N/A	N/A

Figure 7-1 shows how the tools are selected.

The following defined terms are used in Figure 7-1:

Perform Y encoding denotes that the encoder actually goes through the entire encoding process of "using" tool Y in order to determine the encoded frame size.

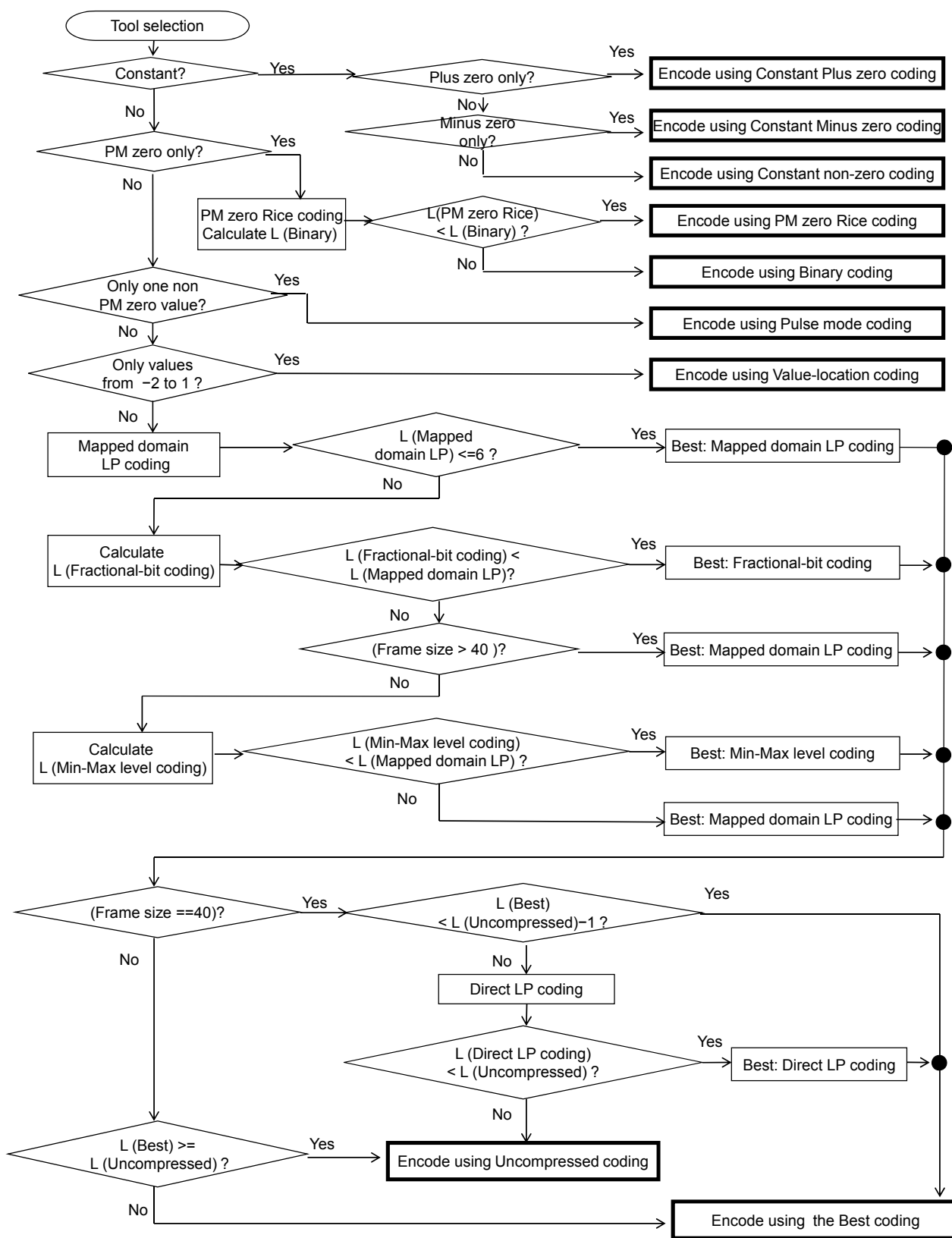
Calculate L(Y) denotes that the encoder does not actually go through the entire process of encoding but rather uses an estimation just to determine the encoded frame size, L(Y), using tool Y.

Encode using Y denotes that the encoder needs to actually perform tool Y, encoding the frame if the encoder HAS NOT previously performed Y encoding; or that the encoder simply uses the encoded bitstream of the frame if the encoder HAS already encoded the frame using tool Y.

L(Best) denotes the smallest encoded bitstream size of the frame for the tools in which the bitstream size has already been calculated.

Best:Y denotes that tool Y is determined to be the best tool which gives the smallest encoded bitstream size of the frame.

If the frame length is zero, prefix code 0x00 is generated and the encoding process is skipped. For other frame lengths, the coder first checks whether all samples in the input frame are all of equal value. If it is the case, one of the constant value coding tools is applied (see clause 7.5). If all values are plus zero 0⁺, then the *Constant Plus zero coding tool* is used. If all values are minus zero 0⁻, then the *Constant Minus zero coding tool* is used. Otherwise, in case of any other constant value, the *Constant non-zero coding tool* is used.



L(Y): length of code compressed by tool Y

Figure 7-1 – Flow chart for encoding tool selection

If the input frame consists of plus zero and minus zero values only, the *Plus-Minus (PM) zero Rice coding tool* (clause 7.6) or the *Binary coding tool* (clause 7.7) is applied (whichever yields better compression). Since the Binary coding tool always produces $(N/8+1)$ octets, only the PM zero Rice coding tool is applied before deciding which of the two tools should be used.

If the input frame consists of plus zero 0^+ and minus zero 0^- values except for one sample, then the *Pulse mode coding tool* (clause 7.8) is used.

Otherwise, the use of *Value-location coding tool* (clause 7.9) is checked. If this fails, the *Mapped domain LP coding tool* (clause 7.10) is carried out. If the encoded data size of the Mapped domain LP coding is larger than 6 octets, two additional tools are tried: the *Fractional-bit coding tool* (if applicable) (clause 7.11), and for the shortest frame length (40), the *Min-Max level coding tool* (clause 7.12). The compressed size of the Fractional-bit coding can be obtained without actually carrying out the compression algorithm, but from the frame length and the values within the frame. Similarly, the Min-Max level coding tool is invoked depending on the values in a frame and the encoding results of other tools. For frame length of 40, if the encoded data size of the tool selected by the above algorithm is not less than the data size of the input ITU-T G.711 frame, then the *Direct LP coding tool* (clause 7.13) is performed, overriding the previously selected tool.

As a final step for all frame lengths, if the resulting encoded data size is found to be larger than the frame length, the *Uncompressed coding tool* (clause 7.4) is selected.

7.4 Uncompressed coding tool

If all tools fail to compress, the encoder produces a prefix for an uncompressed coding tool (e.g., binary representations "--00 0000" for 40-, 80- and 160-sample frames, and "---- 0000" for 240- and 320-sample frames) and simply reproduces the original ITU-T G.711 bitstream after it. In this case, the encoded data size is the input data size plus one octet. Table 7-4 shows the bit-packing format of the uncompressed coding tool.

Table 7-4 – Bit packing format of the uncompressed coding tool

Frame length (N)	40, 80, 160	240, 320
Frame length prefix	2 bits	4 bits
Tool prefix	6 bits	4 bits
G.711 symbols	$N \times 8$ bits	

NOTE – The frame length prefix and the tool prefix are given in Tables 7-1 and 7-2, respectively.

7.5 Constant value coding tools

When all sample values in an input frame are the same, one of the constant value coding tools is applied. As shown in Table 7-2, constant pulse zero values 0^+ , such as 0xd5 for A-law and 0xff for μ -law, are signalled by binary representations "--00 0001" or "---- 0001". Constant minus zero values 0^- , such as 0x55 for A-law and 0x7f for μ -law, are signalled with an octet "--00 0010" or "--- 0010". There are no trailing octets in these cases.

Constant values other than above are signalled by a prefix code, "--00 0011" or "---- 0011", and the prefix octet is followed by the actual value of the constant ITU-T G.711 A-law or μ -law symbol in 8-bit representation, i.e., 1 octet. Table 7-5 shows the bit-packing format of the constant coding tools.

Table 7-5 – Bit packing format of the constant value coding tools

	Constant Plus zero coding and Constant Minus zero coding		Constant non-zero coding	
Frame length (N)	40, 80, 160	240, 320	40, 80, 160	240, 320
Frame length prefix	2 bits	4 bits	2 bits	4 bits
Tool prefix	6 bits	4 bits	6 bits	4 bits
Constant G.711 symbol value	–		8 bits	

7.6 Plus-Minus zero Rice coding tool

When all sample values in an input frame are either plus zero 0^+ or minus zero 0^- , the encoder tries the *Plus-Minus (PM) zero Rice coding tool*.

First of all, this coding tool counts the number of existing plus zero 0^+ and minus zero 0^- samples and detects which value has more occurrences. That value is called more zero 0_m . If the number of occurrences of 0^+ and 0^- is the same, set 0_m to the value 0^+ . The tool prefix "--01 0" or "---- 100" is used for $0_m = 0^+$ and "--011" or "----101" is used for $0_m = 0^-$ (see Table 7-2).

Then, the tool converts the input samples into sequential values of number of running more zero 0_m values followed by a less zero value 0_l , v_i ($0 \leq i \leq N_v - 1$), by the following steps:

- 1) Initialize $i = 0$.
- 2) In order to get an i -th run-length of 0_m s, count the number v_i of contiguous 0_m s followed by a 0_l . If the contiguous 0_m s terminates in the last sample of the frame, the number of contiguous 0_m s should be counted as if a virtual 0_l existed after the last sample.
- 3) Increment i and repeat step 2 until all samples in the frame are processed.
- 4) Set $N_v = i$.

The best Rice parameter value S which gives the minimum code size for those numbers v_i ($0 \leq i \leq N_v - 1$) is selected by examining all possible S values with code length estimation.

The Rice parameter S (S is larger than 0) is encoded with the Huffman code table shown in Table 7-6. The range of S depends on the input frame length.

Finally, the tool encodes the sequence of numbers v_i ($0 \leq i \leq N_v$) using Rice coding, as described in clause 6.9.2, with the best Rice parameter value S .

Table 7-6 – Huffman codes of the Rice parameters for PM zero Rice coding

S	Huffman encodings of S for each frame length			Note
	$N=40$	$N=80$	$N=160, 240, 320$	
1	01	01	01	Note that the value of $S=0$ is not allowed for the PM zero coding tool and code 00 is reserved for the prefix of pulse mode coding.
2	10	10	10	
3	110	1100	1100	
4	1110	1101	1101	
5	1111	1110	11100	
6	–	1111	11101	
7	–	–	11110	
8	–	–	111110	
9	–	–	111111	

Table 7-7 shows the bit-packing format of the Plus-Minus zero Rice coding tool.

Table 7-7 – Bit packing format of the Plus-Minus zero Rice coding tool

Frame length (N)	40, 80	160	240, 320
Frame length prefix	2 bits		4 bits
Tool prefix	3 bits		3 bits
Huffman code for the Rice parameter S	2 to 4 bits	2 to 6 bits	
Run length codes	Variable (Rice codes for runs of more zeros)		
Octet boundary alignment	0 to 7 bits of '0's		

7.7 Binary coding tool

When all sample values in an input frame are either plus zero 0^+ or minus zero 0^- , and if the PM zero Rice coding tool does not reduce the encoded data size, the *Binary coding tool* is applied. This tool generates one-octet prefix code, "--00 0100" or "---- 0100" in binary representation, followed by the input sample values which are converted into one bit per sample. In the generated code, 0 indicates plus zero 0^+ and 1 indicates minus zero 0^- . Table 7-8 shows the bit-packing format of the binary coding tool.

Table 7-8 – Bit packing format of the binary coding tool

Frame length (N)	40, 80, 160	240, 320
Frame length prefix	2 bits	4 bits
Tool prefix	6 bits	4 bits
1 bit per sample	N bits	

7.8 Pulse mode coding tool

When all sample values in a given input frame are either plus zero 0^+ or minus zero 0^- , except one sample, the encoder applies the *Pulse mode coding tool* to the input. The prefix code for the pulse mode coding tool is generated after the prefix code of the frame length identifier.

First of all, this coding tool counts the number of existing plus zero 0^+ and minus zero 0^- samples and detects which one has more occurrences, and the position of the sample with a value being neither 0^+ nor 0^- . The sample, whose value is not 0^+ or 0^- , is called a pulse. 0^+ and 0^- samples are called non-pulse ones. The value which has more occurrences is called more zero 0_m ; the other one is called less zero 0_l . The tool prefix "--01 000" or "---- 1000 0" is used for $0_m = 0^+$ and "--01 100" or "---- 1010 0" is used for $0_m = 0^-$.

After the position and the value of the pulse are stored, the pulse value is considered as a more zero value 0_m . Then the tool converts the input samples into sequential values of the number of running more zero 0_m values followed by a less zero value 0_l , v_i ($0 \leq i \leq N_v - 1$), as described in clause 7.6.

The Rice parameter S is first coded with the Huffman code table shown in Table 7-9. Then, the pulse position index is directly coded according to the input frame length, shown in Table 7-10. To improve the coding efficiency, the tool further compares the differences between the pulse value and more zero 0_m or less zero 0_l value. The more zero 0_m or less zero 0_l with the smaller difference is selected. A 1-bit flag is used to switch between more zero 0_m or less zero 0_l based differential coding. The corresponding difference plus 1 is coded using Rice coding as described in clause 6.9.2, with Rice parameter 0.