INTERNATIONAL TELECOMMUNICATION UNION

CCITT G.727

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

GENERAL ASPECTS OF DIGITAL TRANSMISSION SYSTEMS; TERMINAL EQUIPMENTS

5-, 4-, 3- AND 2-BITS SAMPLE EMBEDDED ADAPTIVE DIFFERENTIAL PULSE CODE MODULATION (ADPCM)

Recommendation G.727

Geneva, 1990

FOREWORD

The CCITT (the International Telegraph and Telephone Consultative Committee) is a permanent organ of the International Telecommunication Union (ITU). CCITT 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 Plenary Assembly of CCITT which meets every four years, establishes the topics for study and approves Recommendations prepared by its Study Groups. The approval of Recommendations by the members of CCITT between Plenary Assemblies is covered by the procedure laid down in CCITT Resolution No. 2 (Melbourne, 1988).

Recommendation G.727 was prepared by Study Group XV and was approved under the Resolution No. 2 procedure on the 14th of December 1990.

CCITT NOTE

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

© ITU 1990

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the ITU.

Recommendation G.727

5-, 4-, 3- AND 2-bits SAMPLE EMBEDDED ADAPTIVE DIFFERENTIAL PULSE CODE MODULATION (ADPCM)

1 Introduction

This Recommendation contains the specification of an embedded Adaptive Differential Pulse Code Modulation (ADPCM) algorithms with 5-, 4-, 3- and 2-bits per sample (i.e., at rates of 40, 32, 24 and 16 kbit/s). The characteristics below are recommended for the conversion of 64 kbit/s. A-law or μ-law PCM channels to/ from variable rate-embedded ADPCM channels.

The Recommendation defines the transcoding law when the source signal is a pulse-code modulation signal at a pulse rate of 64 kbit/s developed from voice frequency analog signals as fully specified by Blue Book Volume, Recommendation G.711.

Applications where the encoder is aware and the decoder is not aware of the way in which the ADPCM codeword bits have been altered, or when both the encoder and decoder are aware of the ways the codewords are altered, or where neither the encoder nor the decoder are aware of the ways in which the bits have been altered can benefit from other embedded ADPCM algorithms.

2 General

The embedded ADPCM algorithms specified here are extensions of the ADPCM algorithms defined in Recommendation G.726 and are recommended for use in packetized speech systems operating according to the Packetized Voice Protocol (PVP) specified in draft Recommendation G.764.

PVP is able to relieve congestion by modifying the size of a speech packet when the need arises. Utilizing the embedded property of the algorithm described here, the least significant bit(s) of each codeword can be disregarded at packetization points and/or intermediate nodes to relieve congestion. This provides for significantly better performance than by dropping packets during congestion.

Section 3 outlines a description of the ADPCM transcoding algorithm. Figure 1/G.727 shows a simplified block diagram of the encoder and the decoder. Sections 4 and 5 provide the principles and functional descriptions of the ADPCM encoding and decoding algorithms, respectively. Section 6 contains the computational details of the algorithm. In this section, each sub-block in the encoder and decoder is precisely defined using one particular logical sequence. If other methods of computation are used, extreme care should be taken to ensure that they yield *exactly* the same value for the output processing variables. Any further departures from the processes detailed in Section 6 will incur performance penalties which may be severe.

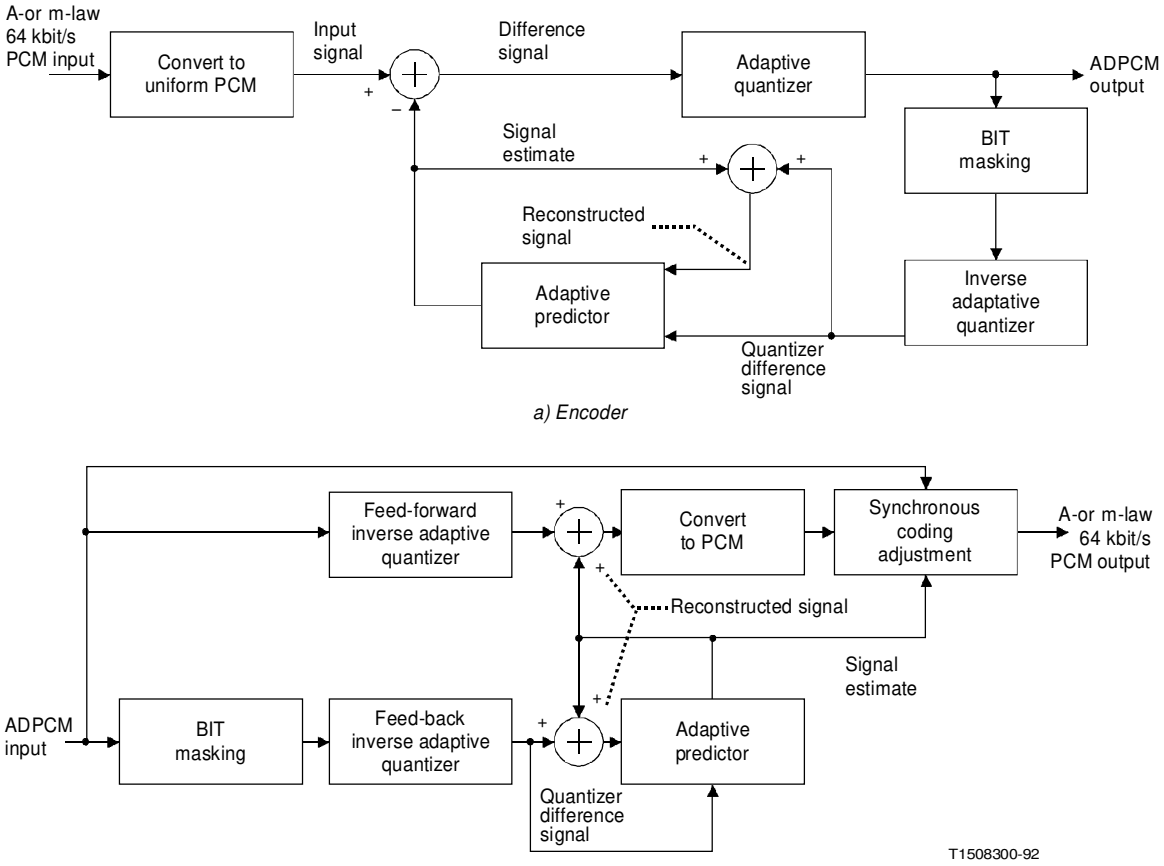

FIGURE 1/G.727

Simplified block diagrams

3 Embedded ADPCM algorithms

Embedded ADPCM algorithms are variable bit rate coding algorthms with the capability of bit dropping outside the encoder and decoder blocks. They consist of a series of algorithms such that the decision levels of the lower rates quantizers are subsets of the quantizer at the highest rate. This allows bit reductions at any point in the network without the need of coordination between the transmitter and the receiver. In contrast, the decision levels of the conventional ADPCM algorithms. such as those in Recommendation G.726, are not subsets of one another and therefore, the transmitter must inform the receiver of the coding rate the encoding algorithm.

Embedded algorithms can accommodate the unpredictable and bursty characteristics of traffic patterns that require congestion relief. Because congestion relief may occur after the encoding is performed, embedded coding is different from variable-rate coding where the encoder and decoder must use the same number of bits in each sample. In both cases, the decoder must be told the number of bits to use in each sample.

Embedded algorithms produce code words which contain enhancement bits and core bits. The Feed-Forward (FF) path utilizes enhancement and core bits, while the Feedback (FB) path uses core bits only. The inverse quantizer and the predictor of both the encoder and the decoder use the core bits. With this structure, enhancement bits can be discarded or dropped during network congestion. However, the number of core bits in the FB paths of both the encoder and decoder must remain the same to avoid mistracking.

The four embedded ADPCM rates are 40, 32, 24 and 16 kbit/s, where the decision levels for the 32, 24 and 16 kbit/s quantizers are sub-sets of those for the 40 kbit/s quantizer. Embedded ADPCM algorithms are referred to by (*x, y*) pairs where *x* refers to the FF (enhancement and core) ADPCM bits and *y* refers to the FB (core) ADPCM bits. For example, if *y* is set to 2 bits, (5,2) will represent the 40 kbits/s embedded algorithm, (4,2) will represent the 32 kbit/s embedded algorithm, (3,2) will represent the 24 kbit/s embedded algorithm and (2,2) the 16 kbit/s algorithm. The bit rate is never less than 16 kbit/s because the minimum number of core bits is 2. Simplified block diagrams of both the embedded ADPCM encoder and decoder are shown in Figure 1/G.727.

The Recommendation provides coding rates of 40, 32, 24 and 16 kbit/s and core rates of 32, 24 and 16 kbit/s. This corresponds to the following pairs: (5,2), (4,2), (3,2), (2,2); (5,3), (4,3), (3,3); (5,4), (4,4).

3.1 *ADPCM encoder*

Subsequent to the conversion of the A-law or μ-law PCM input signal to uniform PCM, a difference signal is obtained by subtracting an estimate of the input signal from the input signal itself. An adaptive 4-, 8-, 16- or 32-level quantizer is used to assign 2, 3, 4 or 5 binary digits to the value of the difference signal for transmission to the decoder. (Not all the bits necessarily arrive at the decoder since some of these bits can be dropped to relieve congestion in the packet network. For a given received sample, however, the core bits are guaranteed arrival if there are no transmission errors and the packets arrive at destination.) FB bits are fed to the inverse quantizer. The number of core bits depends on the embedded algorithm selected. For example, the (5,2) algorithm will always contain 2 core bits. The inverse quantizer produces a quantized difference signal from these binary digits. The signal estimate is added to this quantized difference signal to produce the reconstructed version of the input signal. Both the reconstructed signal and the quantized difference signal are operated upon by an adaptive predictor which produces the estimate of the input signal, thereby completing the feedback loop.

3.2 *ADPCM decoder*

The decoder includes a structure identical to the FB portion of the encoder. In addition, there is also an FF path that contains a uniform PCM to A-law or μ-law conversion. The core as well as the enhancement bits are used by the synchronous coding adjustment block to prevent cumulative distortion on synchronous tandem codings (ADPCM-PCM-ADPCM, etc., digital connections) under certain conditions (see § 5.10). The synchronous coding adjustment is achieved by adjusting the PCM output codes to eliminate quantizing distorsion in the next ADPCM encoding stage.

3.3 *One's density requirements*

These algorithms produce the all-zero code words. If requirements on one's density exist in national networks, other methods should be used to ensure that this requirement is satisfied.

3.4 *Applications*

In the anticipated application with G.764, the Coding Type (CT) field and the block Dropping Indicator (BDI) field in the packet header defined in G.764 will inform the coder of what algorithm to use. For all other applications, the information that PVP supplies must be made known to the decoder.

4 ADPCM encoder principles

Figure 2/G.727 is a block schematic of the encoder. For each variable to be described, *k i*s the sampling index and samples are taken at 125 µs intervals. A description of each block is given in §§ 4.1 to 4.9.

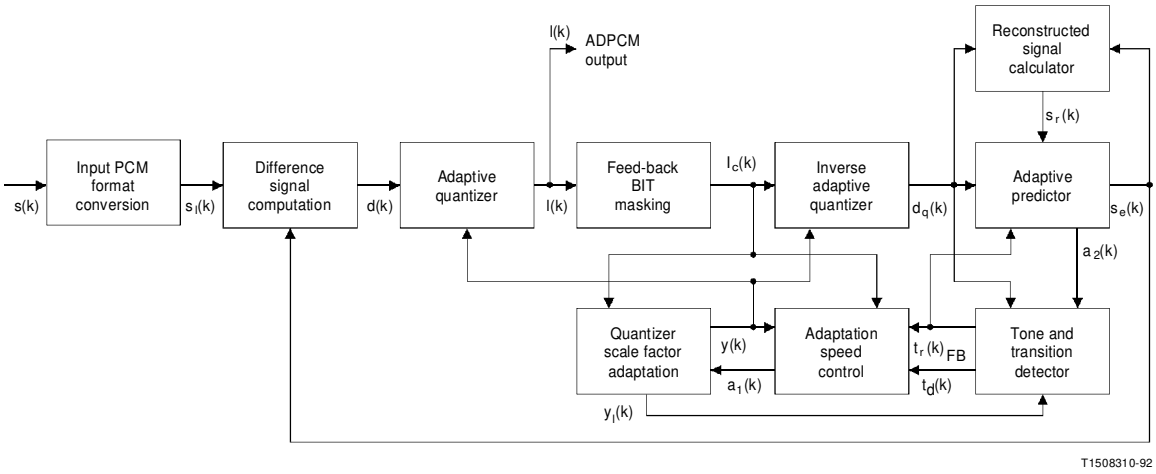

FIGURE 2/G.727

Encoder block schematic

4.1 *Input PCM format conversion*

This block converts the input signal $s(k)$ from A-law or µ-law PCM to a uniform PCM signal $s_l(k)$.

4.2 *Difference signal computation*

This block calculates the difference signal $d(k)$ from the uniform PCM signal $s_l(k)$ and the signal estimate $s_e(k)$.

$$d(k) = s_l(k) - s_e(k) \quad (4\text{-}1)$$

4.3 *Adaptive quantizer*

A 4-, 8-, 16- or 32-level non-uniform mid-rise adaptive quantizer is used to quantize the difference signal $d(k)$. Prior to quantization, $d(k)$ is converted to a base 2 logarithmic representation and scaled by $y(k)$ which is computed by the scale factor adaptation block. The normalized input/output characteristic (infinite precision values) of the quantizer is given in Tables 1/G.727 through 4/G.727 for the 16, 24, 32 and 40 kbit/s algorithms, respectively. Two, three, four or five binary digits are used to specify the quantized level representing $d(k)$ (the most significant bit represents the sign bit and the remaining bits represent the magnitude). The 2-, 3-, 4- or 5-bit quantizer output $I(k)$ forms the 16, 24, 32 or 40 kbit/s output signal and is also fed to the bit-masking block. $I(k)$ includes both the enhancement and core bits.

4.4 *Bit masking*

This block produces the core bits $I_c(k)$ by logically right-shifting the input signal $I(k)$ so as to mask the maximum droppable (least significant) bits. The number of bits to mask and the number of places to right shift depend on the embedded algorithm selected. For example, this block will mask the two least significant bits (LSB's) and shift the remaining bits two places to the right when the (4,2) algorithm is selected. The output of the bit-masking block $I_c(k)$ is fed to the inverse adaptive quantizer, the quantizer scale factor adaptation and the adaptation speed control blocks.

TABLE 1/G.727

Quantizer normalized input/output
Characteristic for 16 kbit/s embedded operation

Normalized quantizer input range $\log_2 \lvert d(k) \rvert - y(k)$	$\lvert I(k) \rvert$ $\lvert I_c(k) \rvert$	Normalized quantizer output $\log_2 \lvert d_q(k) \rvert - y(k)$
(–∞, 2.04)	0	0.91
[2.04, ∞)	1	2.85

Note – In Tables 1/G.727 through 4/G.727, "[" indicates that the endpoint value is included in the range, and "(" or ")" indicates that the endpoint value is excluded from the range.

TABLE 2/G.727

Quantizer normalized input/output
Characteristic for 24 kbit/s embedded operation

Normalized quantizer input range $\log_2 \lvert d(k) \rvert - y(k)$	$\lvert I(k) \rvert$ $\lvert I_c(k) \rvert$	Normalized quantizer output $\log_2 \lvert d_q(k) \rvert - y(k)$
(–∞, 0.96)	0	–0.09
[0.96, 2.04)	1	1.55
[2.04, 2.78)	2	2.40
[2.78, ∞)	3	3.09

TABLE 3/G.727

Quantizer normalized input/output
Characteristic for 32 kbit/s embedded operation

Normalized quantizer input range $\log_2 \lvert d(k) \rvert - y(k)$	$\lvert I(k) \rvert$ $\lvert I_c(k) \rvert$	Normalized quantizer output $\log_2 \lvert d_q(k) \rvert - y(k)$
$(-\infty, -0.05)$	0	−1.06
[−0.05, 0.96)	1	0.53
[0.96, 1.58)	2	1.29
[1.58, 2.04)	3	1.81
[2.04, 2.42)	4	2.23
[2.42, 2.78)	5	2.59
[2.78, 3.16)	6	2.95
[3.16, ∞)	7	3.34

TABLE 4/G.727

Quantizer normalized input/output
Characteristic for 40 kbit/s embedded operation

Normalized quantizer input range $\log_2 \lvert d(k) \rvert - y(k)$	$\lvert I(k) \rvert$	Normalized quantizer output $\log_2 \lvert d_q(k) \rvert - y(k)$
$(-\infty, -1.05)$	0	−2.06
[−1.05, −0.05)	1	−0.48
[−0.05, 0.54)	2	0.27
[0.54, 0.96)	3	0.76
[0.96, 1.30)	4	1.13
[1.30, 1.58)	5	1.44
[1.58, 1.82)	6	1.70
[1.82, 2.04)	7	1.92
[2.04, 2.23)	8	2.13
[2.23, 2.42)	9	2.33
[2.42, 2.60)	10	2.51
[2.60, 2.78)	11	2.69
[2.78, 2.97)	12	2.87
[2.97, 3.16)	13	3.05
[3.16, 3.43)	14	3.27
[3.43, ∞)	15	3.56

4.5 *Inverse adaptive quantizer*

The inverse quantizer uses the core bits to compute a quantized version $d_q(k)$ of the difference signal using the scale factor $y(k)$ and Table 1/G.727, 2/G.727, 3/G.727 or 4/G.727 and then taking the antilog to the base 2 of the result. The estimated difference $s_e(k)$ is added to $d_q(k)$ to produce the reconstructed version $s_r(k)$ of the input signal. Table 1/G.727, 2/G.727, 3/G.727 or 4/G.727 will be applicable only when there are 2, 3, 4 or 5 bits, respectively, in the FF path.

4.6 *Quantizer scale factor adaptation*

This bloc computes $y(k)$, the scaling factor for the quantizer and the inverse quantizer. (The scaling factor $y(k)$ is also fed to the adaptation speed control block.) The inputs are the bit-masked output $I_c(k)$ and the adaptation speed control parameter $a_l(k)$.

The basic principle used in scaling the quantizer is bimodal adaptation:

— fast for signals (e.g., speech) that produce difference signals with large fluctuations,

— slow for signals (e.g., voiceband data, tones) that produce difference signals with small fluctuations.

The speed of adaptation is controlled by a combination of fast and slow scale factors.

The fast (unlocked) scale factor $y_u(k)$ is recursively computed in the base 2 logarithmic domain from the resultant logarithmic scale factor $y(k)$:

$$y_u(k) = (1 - 2^{-5})\, y(k) + 2^{-5}\, W[I_c(k)] \tag{4-2}$$

where

$y_u(k)$ is limited by $1.06 \leq y_u(k) \leq 10.00$.

For 2-core-bit operation (1 sign bit), the discrete function $W[I_c(k)]$ is defined as follows (infinite precision values):

$\lvert I_c(k) \rvert$	1	0
$W[I_c(k)]$	27.44	−1.38

For 3-core-bit operation (1 sign bit), the discrete function $W[I_c(k)]$ is defined as follows (infinite precision values):

$\lvert I_c(k) \rvert$	3	2	1	0
$W[I_c(k)]$	36.38	8.56	1.88	–0.25

For 4-core-bit (1 sign bit) operation, the discrete function $W[I_c(k)]$ is defined as follows (infinite precision values):

$\lvert I_c(k) \rvert$	7	6	5	4	3	2	1	0
$W[I_c(k)]$	69.25	21.25	11.50	6.13	3.13	1.69	0.25	–0.75

The factor $(1 - 2^{-5})$ introduces finite memory into the adaptive process so that the states of the encoder and decoder converge following transmission errors.

The slow (locked) scale factor $y_l(k)$ is derived from $y_u(k)$ with a low pass filter operation:

$$y_l(k) = (1 - 2^{-6})\, y_l(k - 1) + 2^{-6}\, y_u(k) \tag{4-3}$$

The fast and slow scale factors are then combined to form the resultant scale for:

$$y(k) = a_l(k)\, y_u(k - 1) + [1 - a_l(k)]\, y_l(k - 1) \tag{4-4}$$

where

$0 \le a_l(k) \le 1$.

4.7 *Adaptation speed control*

The controlling parameter $a_l(k)$ can assume values in the range [0, 1]. It tends towards unity for speech signals and towards zero for voiceband data signals. It is derived from a measure of the rate-of-change of the difference signal values.

Two measures of the average magnitude of $I_c(k)$ are computed:

$$d_{ms}(k) = (1 - 2^{-5})\, d_{ms}(k-1) + 2^{-5}\, F\,[I_c(k-1)] \tag{4-5}$$

and

$$d_{ml}(k) = (1 - 2^{-7})\, d_{ml}(k-1) + 2^{-7}\, F\,[I_c(k-1)] \tag{4-6}$$

where

$F[I_c(k)]$ is defined by

$\mid I_c(k) \mid$	1	0
$F[I_c(k)]$	7	0

for 2-core-bit (1 sign bit) operation; or

$\mid I_c(k) \mid$	3	2	1	0
$F[I_c(k)]$	7	2	1	0

for 3-core-bit (1 sign bit) operation; or

$\mid I_c(k) \mid$	7	6	5	4	3	2	1	0
$F[I_c(k)]$	7	3	1	1	1	0	0	0

for 4-core-bit (1 sign bit) operation.

Thus, $d_{ms}(k)$ is a relatively short term average of $F[I_c(k)]$ and $d_{ml}(k)$ is a relatively long term average of $F[I_c(k)]$.

Using these two averages, the variable $a_p(k)$ is defined:

$$a_p(k) = \begin{cases} (1 - 2^{-4})\, a_p(k-1) + 2^{-3}, \text{ if } \mid d_{ms}(k) - d_{ml}(k) \mid \geq 2^{-3}\, d_{ml}(k) \\ (1 - 2^{-4})\, a_p(k-1) + 2^{-3}, \text{ if } y(k) < 3 \\ (1 - 2^{-4})\, a_p(k-1) + 2^{-3}, \text{ if } t_d(k) = 1 \\ 1, \text{ if } t_r(k) = 1 \\ (1 - 2^{-4})\, a_p(k-1), \text{ otherwise} \end{cases} \tag{4-7}$$

Thus, $a_p(k)$ tends towards the value 2 if the difference between $d_{ms}(k)$ and $d_{ml}(k)$ is large (average magnitude $I_c(k)$ changing), for an idle channel [indicated by $y(k) < 3$] or for partial band (indicated by $t_d(k) = 1$ as described in § 4.9) signals. The value of $a_p(k)$ tends towards the value 0 if the difference is small [average magnitude of $I_c(k)$ relatively constant]. Note that $a_p(k)$ is set to one upon detection of a partial band signal transition [indicated by $t_r(k) = 1$, see § 4.9].

$a_p(k-1)$ is then limited to yield $a_l(k)$ used in equation (4-4) above:

$$a_l(k) = \begin{cases} 1, & a_p(k-1) > 1 \\ a_p(k-1), & a_p(k-1) \leq 1 \end{cases} \tag{4-8}$$

This asymmetrical limiting has the effect of delaying the start of a fast to slow state transition until the absolute value of $I_c(k)$ remains constant for some time. This tends to eliminate premature transitions for pulsed input signals such as switched carrier voiceband data.

4.8 *Adaptive predictor and feedback reconstructed signal calculator*

The primary function of the adaptive predictor is to compute the signal estimate $s_e(k)$ from the quantized difference signal $d_q(k)$. Two adaptive predictor structures are used, a sixth order section that models zeroes and a second order section that models poles in the input signal. This dual structure effectively caters for the variety of input signals which might be encountered.

The signal estimate is computed by:

$$s_e(k) = \sum_{i=1}^{2} a_i(k-1)\, s_r(k-i) + s_{ez}(k) \tag{4-9}$$

where

$$s_{ez}(k) = \sum_{i=1}^{6} b_i(k-1)\, d_q(k-i)$$

and the reconstructed signal is defined as

$$s_r(k-i) = s_e(k-i) + d_q(k-i)$$

Both sets of predictor coefficients are updated using a simplified gradient algorithm.

For the second order predictor:

$$a_1(k) = (1 - 2^{-8})\, a_1(k-1) + (3 \cdot 2^{-8})\, \text{sgn}\,[p(k)]\, \text{sgn}\,[p(k-1)] \tag{4-10}$$

$$\begin{aligned} a_2(k) = (1,- 2^{-7})\, a_2(k-1) + 2^{-7} \{ & \text{sgn}\,[p(k)]\, \text{sgn}\,[p(k-2)] \\ & - f\,[a_1(k-1)]\, \text{sgn}\,[p(k)]\, \text{sgn}\,[p(k-1)] \} \end{aligned} \tag{4-11}$$

where

$$p(k) = d_q(k) + s_{ez}(k)$$

$$f(a_1) = \begin{cases} 4a_1, & |\, a_1 \,| \leq 2^{-1} \\ 2\text{sgn}(a_1), & |\, a_1 \,| > 2^{-1} \end{cases}$$

and sgn [0] = 1, except sgn $[p(k - i)]$ is defined to be 0 only if $p(k - i) = 0$ and $i = 0$; with the stability constraints;

$$|\, a_2(k) \,| \leq 0.75 \text{ and } |\, a_1(k) \,| \leq 1 - 2^{-4} - a_2(k)$$

If $t_r(k) = 1$ (see § 4.9), then $a_1(k) = a_2(k) = 0$.

For the sixth order predictor:

$$b_i(k) = (1 - 2^{-8})\, b_i(k - 1) + 2^{-7} \text{ sgn } [d_q(k)] \text{ sgn } [d_q(k - i)] \qquad (4\text{-}12)$$

for $i = 1, 2, \ldots, 6$.

If $t_r(k) = 1$ (see § 4.9), then $b_1(k) = b_2(k) = ... = b_6(k) = 0$.

As above, sgn [0] = 1, except sgn $[d_q(k) - i)]$ is defined to be 0 only if $d_q(k - i) = 0$ and $i = 0$. Note that $b_i(k)$ is implicitly limited to ± 2.

4.9 *Tone and transition detector*

In order to improve performance for signals originating from Frequency Shift Keying (FSK) modems operating in the character mode, a two-step detection process is defined. First, partial band signal (e.g. tone) detection is invoked to that the quantizer can be driven into the fast mode of adaptation:

$$t_d(k) = \begin{cases} 1, & a_2(k) < -0.71875 \\ 0, & \text{otherwise} \end{cases} \qquad (4\text{-}13)$$

In addition, a transition from a partial band signal is defined so that the predictor coefficients can be set to zero and the quantizer can be forced into the fast mode of adaptation:

$$t_r(k) = \begin{cases} 1, & a_2(k) < -0.71875 \text{ and } |\, d_q(k) \,| > 24 \cdot 2^{yl(k)} \\ 0, & \text{otherwise} \end{cases} \qquad (4\text{-}14)$$

5 ADPCM decoder principles

Figure 3/G.727 is a block schematic of the decoder. A functional description of each block is given in §§ 5.1 to 5.10 below. There is an FB path and an FF path. The FB path uses the core bits to calculate the signal estimate. The FF path contains the core and enhanced bits and reconstructs the output PCM code word.

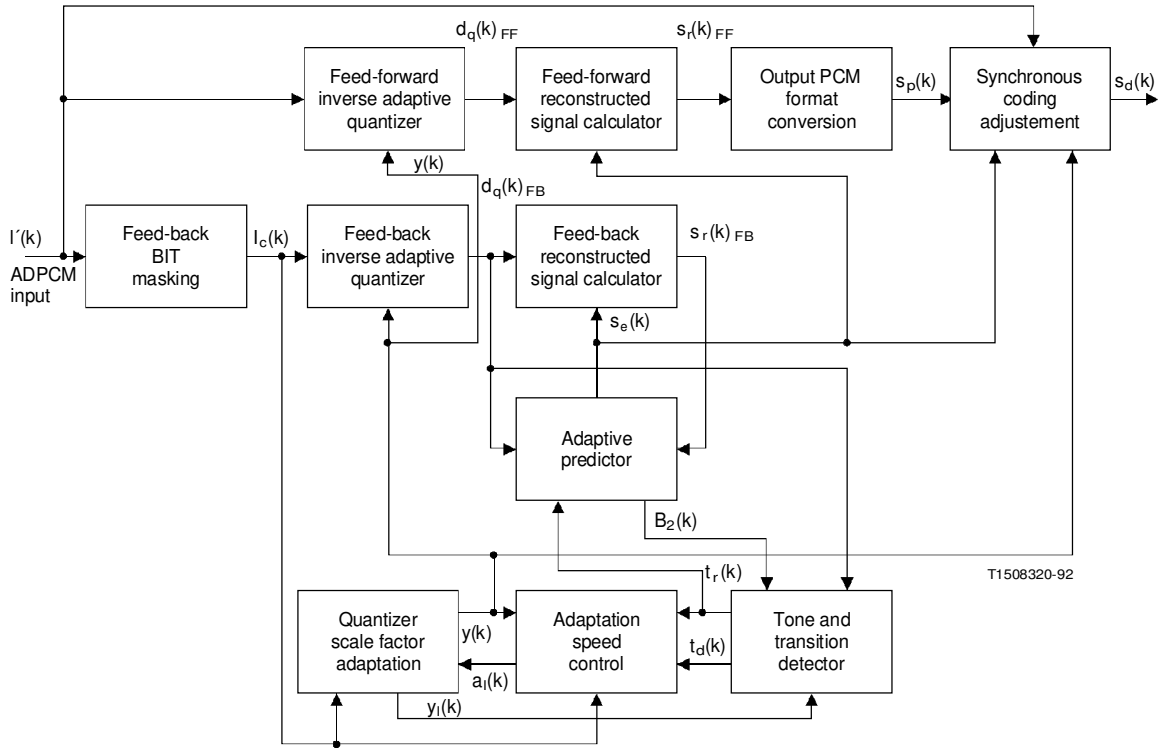

FIGURE 3/G.727

Decoder block schematic

5.1 *Bit masking*

The function of this block is described in § 4.4.

5.2 *Feedback inverse adaptive quantizer*

The function of this block is described in § 4.5.

5.3 *Feed -forward inverse adaptive quantizer*

A quantized difference signal $d_q(k)$ of the difference signal is produced from the input $I'(k)$ which contains the core and enhanced bits.

5.4 *Quantizer scale factor adaptation*

The function of this block is described in § 4.6.

5.5 *Adaptation speed control*

The function of this block is described in § 4.7.

5.6 *Adaptive predictor and feedback reconstructed signal calculator*

The functions of this block are described in § 4.8.

5.7 *Feed-forward reconstructed signal calculator*

This block receives input from the FF inverse adaptive quantizer and the adaptive predictor to reconstruct the uniform PCM signal that is fed to the PCM format conversion block.

5.8 *Tone and transition detector*

The function of this block is described in § 4.9.

5.9 *Output PCM format conversion*

This block converts the reconstructed uniform PCM signal $s_r(k)_{FF}$ into an A-law or μ-law PCM signal $s_p(k)$ as required.

5.10 *Synchronous coding adjustment*

The synchronous coding adjustment prevents cumulative distorsion occurring on synchronous tandem codings (ADPCM-PCM-ADPCM, etc., digital connections) for:

– error-free transmission,

– instances when the embedded ADPCM and intermediate 64 kbit/s PCM bit streams are not disturbed by digital signal processing devices.

If the coder and decoder have different initial conditions, as may occur after switching for example, then the synchronous tandeming may take time to establish. Furthermore, if this property is disturbed or not acquired initially, then it may be recovered for those signals of sufficient level with spectra that occupy the majority of the 200 to 3400 Hz band (e.g., speech, 4800 bit/s voiceband data).

When a decoder is synchronously connected to an encoder, the synchronous coding adjustment block estimates quantization in the encoder. If all state variables in both the decoder and encoder have identical values and there are no transmission errors, the forced equivalence of both feed-forward quantizer output sequences for all values of k guarantees the non-accumulation of distortion.

This is accomplished by first converting the A-law or μ-law signal $s_p(k)$ to a uniform PCM signal $s_{tx}(k)$ and then computing a difference signal $d_x(k)$:

$$d_x(k) = s_{lx}(k) - s_e(k) \tag{5-1}$$

The difference signal $d_x(k)$ is then examined to see if it falls in the ADPCM quantizer decision interval determined by $I(k)$ and $y(k)$. The signal $s_d(k)$ is then defined as follows:

$$s_d(k) = \begin{cases} s^+_p(k), & d_x(k) < \text{lower interval boundary} \\ s^-_p(k), & d_x(k) \geq \text{upper interval boundary} \\ s_p(k), & \text{otherwise} \end{cases} \quad (5\text{-}2)$$

where

$s_d(k)$: the output PCM code word of the decoder;

$s^+_p(k)$: the PCM code word that represents the next more positive PCM output level (when $s_p(k)$ represents the most positive output level, then $s^+_p(k)$ is constrained to be the value $s_p(k)$;

$s^-_p(k)$: the code word that represents the next more negative PCM output level (when $s_p(k)$ represents the most negative output level, then $s^-_p(k)$: is constrained to be the value $s_p(k)$.

6 Computational details

Computational details for each of the encoder and decoder elements are provided in §§ 6.1 and 6.2.

Proper timing for the encoder and decoder is obtained by executing all of the delay blocks simultaneously and proceeding to calculate the signals which can be derived using this information. For example, the Signal Estimate (SE) of Figure 10/G.727 is calculated using delay values and then SE is used as shown in Figure 4/G.727.

6.1 *Input and output signals*

Table 5/G.727 defines the input and output signals for the encoder and decoder. Table 6/G.727 defines control variables for the algorithm.

An optional signal R represents a reset function that sets all internal memory elements to a specified condition so that an encoder or decoder can be forced into a known state. For applications that require an immediate reset function (e.g. digital circuit multiplication equipment), the reset is mandatory, not optional.

TABLE 5/G.727

Input and output signals

ENCODER			
	Name	Number of bits	Description
Input	S	8	PCM input word
Input	LAW	1	PCM law select, 0 = μ-law, 1 = A-law
Input	R (optional)	1	Reset
Output	I	E + C	C = core bits, E = Enhancement bits
DECODER			
	Name	Number of bits	Description
Input	I′	C + E	C = core bits, E = Enhancement bits
Input	LAW	1	PCM law select, 0 = μ-law, 1 = A-law
Input	R (optional)	1	Reset
Output	SD	8	Decoder PCM output word

TABLE 6/G.727

Control variables

Coding rate kbit/s	Core bits (C)	Enhancement bits (E)
16	2	0
24	2	1
	3	0
32	2	2
	3	1
	4	0
40	2	3
	3	2
	4	1

Note – For 16-kbit/s, 24-kbit/s and 32 kbit/s operation, E = 0 corresponds to the algorithm at higher rates with all the enhancement bits dropped. For 40-kbit/s operation, E = 0 is invalid because no higher rate embedded algorithm has been defined.

TABLE 7/G.727

Internal processing variables

Name	Bits	Binary representation	Optional reset values	Description
A1[a], A2[a]	16 TC	S,0,..,–14	0	Delayed predictor second order coefficients
A1P, A2P	16 TC	S,0,..,–14		Second order predictor coefficients
A1R, A2R	16 TC	S,0,..,–14		Triggered second order predictor coefficients
A1T	16 TC	S,0,..,–14		Unlimited a_1 coefficient
A2T	16 TC	S,0,..,–14		Unlimited a_2 unlimited
AL	7 SM	0,..,–6		Limited speed control parameter
AP[a]	10 SM	1,..,–8	0	Delayed unlimited speed control parameter
APP	10 SM	1,..,–8		Unlimited speed control parameter
APR	10 SM	1,..,–8		Triggered unlimited speed control parameter
AX	1 SM	1		Speed control parameter update
B1[a],...,B6[a]	16 TC	S,0,..,–14	0	Delayed sixth order predictor coefficients
B1P,...,B6P	16 TC	S,0,..,–14		Sixth order predictor coefficients
B1R,...,B6R	16 TC	S,0,..,–14		Triggered sixth order predictor coefficients
D	16 TC	S,14,..,0		Difference signal, only in encoder
DL	11 SM	3,..,–7		Log_2 (difference signal), only in encoder
DLN	12 TC	S,3,..,–7		Log_2 (normalized difference), only in encoder
DLNX	12 TC	S,3,..,–7		Log_2 (normalized difference), only in decoder
DLX	11 SM	3,..,–7		Log_2 (difference signal), only in decoder
DML[a]	14 SM	2,..,–11	0	Delayed long term average of F(I) sequence
DMLP	14 SM	2,..,–11		Long term average of F(I) sequence
DMS[a]	12 SM	2,..,–9	0	Delayed short term average of F(I) sequence
DMSP	12 SM	2,..,–9		Short term average of F(I) sequence
DQ_{FB}	15 SM	S,13,..,0		Feed-back (FB) quantized difference signal
DQ_{FF}	15 SM	S,13,..,0		Feed-forward (FF) quantized difference signal
DQ0	11 FL	S,4e,6m		Quantized difference signal with delay 0
DQ1[a],...,DQ6[a]	11 FL	S,4e,6m	32	Quantized difference signal with delays 1 to 6
DQL_{FB}	12 TC	S,3,..,–7		Log_2 (FB quantized difference signal)
DQL_{FF}	12 TC	S,3,...,–7		Log_2 (FF quantized difference signal)
$DQLN_{FB}$	12 TC	S,3,...,–7		Log_2 (FB normalized quantized difference)
$DQLN_{FF}$	12 TC	S,3,..,–7		Log_2 (FF normalized quantized difference)
DQS_{FB}	1 TC	S		Sign bit of FB quantized difference signal
DQS_{FF}	1 TC	S		Sign bit of FF quantized difference signal

TABLE 7/G.727 *(Cont.)*

Name	Bits	Binary representation	Optional reset values	Description
DS	1 TC	S		Sign bit of difference signal, only in encoder
DSX	1 TC	S		Sign bit of difference, signal, only in decoder
DX	16 TC	S,14,..,0		Difference signal, only in decoder
FI	3 SM	2,..,0		Output of F(I)
I_c	2 SM	S,0		2 bit core ADPCM bits
I_c	3 SM	S,1,0		3 bit core ADPCM bits
I_c	4 SM	S,2,..,0		4 bit core ADPCM bits
PK0	1 TC	S		Sign of DQ + SEZ with delay 0
PK1[a], PK2[a]	1 TC	S	0	Sign of DQ + SEZ with delays 1 and 2
SE	15 TC	S,13,..,0		Signal estimate
SEZ	15 TC	S,13,..,0		Sixth order predictor partial signal estimate
SIGPK	1 TC	0		sgn [$p(k)$] flag
SL	14 TC	S,12,..,0		Linear input signal, only in encoder
SLX	14 TC	S,12,..,0		Quantized reconstructed, signal, only in decoder
SP	8			PCM reconstructed signal, only in decoder
SR_{FF}	16 TC	S,14,..,0		Reconstructed signal
SR_{FB}	16 TC	S,14,...,0		Reconstructed signal
SR0	11 FL	S,4e,6m		Reconstructed signal with delay 0
SR1[a], SR2[a]	11 FL	S,4e,6m	32	Reconstructed signal with delays 1 and 2
TD[a]	1 TC	0	0	Delayed tone detect
TDP	1 TC	0		Tone detect
TDR	1 TC	0		Triggered tone detect
TR	1 TC	0		Transition detect
U1,...,U6	1 TC	S		Sixth order predictor coefficient update sign bit
WA1,WA2	16 TC	S,13,..,–1		Partial product of signal estimate
WB1,...,WB6	16 TC	S,13,..,–1		Partial product of signal estimate
WI	12 TC	S,6,..,–4		Quantizer multiplier
Y	13 SM	3,..,–9		Quantizer scale factor
YL[a]	19 SM	3,..,–15	34816	Delayed slow quantizer scale factor
YLP	19 SM	3,..,–15		Slow quantizer scale factor
YU[a]	13 SM	3,..,–9	544	Delayed fast quantizer scale factor
YUP	13 SM	3,..,–9		Fast quantizer scale factor
YUT	13 SM	3,..,–9		Unlimited fast quantizer scale factor

a) Indicates variables that are set to specific values by the reset. When reset is invoked, the output of the DELAY sub-block (see § 7.2.5) is given in column 4.

TC Two's complement
SM Signed magnitude
FL Floating point
e Exponent bits
m Mantissa bits
S Sign bit

6.2 *Description of variables and detailed specification of sub-blocks*

This section contains a detailed expansion off all blocks in Figures 2/G.727 and 3/G.727, described in §§ 4 and 5. The expansions are illustrated in Figures 4/G.727 to 13/G.727 with the internal processing variables as defined in Table 7/G.727. A brief functional description and full specification is given for each sub-block.

The notations used in the sub-block descriptions are as follows:

<<*n* *n*-bit shift left operation (zero fill);

>>*n* *n*-bit shift right operation (in the direction of the least significant bit and zero fill);

& Logical "and" operation;

+ Arithmetic addition;

– Arithmetic subtraction;

* Arithmetic multiplication;

** Logical "exclusive or" operation;

|
| Comments to equations.
|

6.2.1 *Input PCM format conversion and difference signal computation*

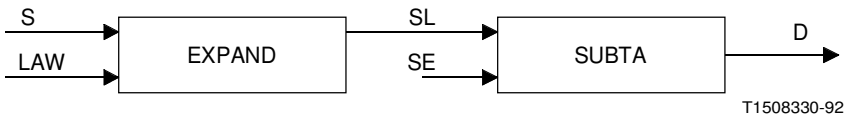

FIGURE 4/G.727

Input PCM format conversion and difference signal computation

EXPAND

Input: S (SP in decoder), LAW

Output: SL (SLX in decoder)

Function: Convert either A-law or μ-law PCM to uniform PCM.

Decode PCM code word, S, according to Recommendation G.711 using character signals (column 6, before inversion of even bits for A-law) and values at decoder output (column 7). The values at decoder output, SS, must be represented in 13-bit signed magnitude form for A-law PCM and 14-bit signed magnitude form for μ-law PCM (the sign bit is equal to one for negative values).

Note – For A-law S (and SP) includes even bit inversion (see Note 2 below Table 1/G.711).

when $LAW = 0$, $SSS = SS >> 13$	μ-law
$SSQ = SS \,\&\, 8191$	
when $LAW = 1$, $SSS = SS >> 12$	
$SSM = SS \,\&\, 4095$	A-law
$SSQ = SSM << 1$	

then

$$SL = \begin{cases} SSQ, & SSS = 0 \\ (16384 - SSQ) \,\&\, 16383, & SSS = 1 \end{cases}$$

Convert signed magnitude to two's complement

SUBTA

Inputs: SL (SLX in decoder), SE

Output: D (DX in decoder)

Function: Compute difference signal by subtracting signal estimate from input signal (or quantized reconstructed signal in decoder).

$SLS = SL >> 13$

$$SLI = \begin{cases} SL, & SLS = 0 \\ 49152 + SL, & SLS = 1 \end{cases}$$ | Sign extension

$SES = SE >> 14$

$$SEI = \begin{cases} SE, & SES = 0 \\ 32768 + SE, & SES = 1 \end{cases}$$ | Sign extension

$D = (SLI + 65536 - SEI)\ \&\ 65535$

6.2.2 *Adaptive quantizer*

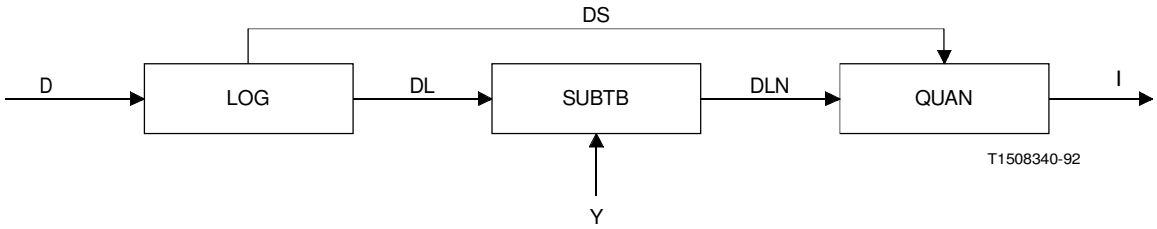

FIGURE 5/G.727

Adaptive quantizer

LOG

Input: D (DX in decoder)

Outputs: DL (DLX in decoder), DS (DSX in decoder)

Function: Convert difference signal from the linear to the logarithmic domain.

$DS = D >> 15$

$$DQM = \begin{cases} D, & DS = 0 \\ (65536 - D)\ \&\ 32767, & DS = 1 \end{cases}$$

| Convert D from two's
| complement to signed
| magnitude

$$EXP = \begin{cases} 14, & 16384 \le DQM \\ 13, & 8192 \le DQM \le 16383 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 1, & 2 \le DQM \le 3 \\ 0, & 0 \le DQM \le 1 \end{cases}$$

| Compute exponent

$MANT = ((DQM << 7) >> EXP)\ \&\ 127$

| Compute approximation
| $\log_2 (1 + x) = x$

$DL = (EXP << 7) + MANT$

| Combine 7 mantissa bits and
| 4 exponent bits into one
| 11-bit word

QUAN (encoder only)

Inputs: DLN, DS

Output: I

Function: Quantize difference signal in logarithmic domain.

TABLE 8/G.727

16 kbit/s quantizer decision levels and 2-bit outputs

DS	DLN	I	
		12	
0	261-2047	01	
0	0- 260	00	– –| Positive portion of interval
0	2048-4095	00	– –| Negative portion of interval
1	2048-4095	11	– –| Negative portion of interval
1	0- 260	11	– –| Positive portion of interval
1	261-2047	10	

TABLE 9/G.727

24 kbit/s quantizer decision levels and 3-bit outputs

DS	DLN	I	
		123	
0	356-2047	011	
0	261- 355	010	
0	123- 260	001	
0	0- 122	000	– –| Positive portion of interval
0	2048-4095	000	– –| Negative portion of interval
1	2048-4095	111	– –| Negative portion of interval
1	0- 122	111	– –| Positive portion of interval
1	123- 260	110	
1	261- 355	101	
1	356-2047	100	

TABLE 10/G.727

32 kbit/s quantizer decision levels and 4-bit outputs

DS	DLN	I	
		1234	
0	405-2047	0111	
0	356- 404	0110	
0	310- 355	0101	
0	261- 309	0100	
0	202- 260	0011	
0	123- 201	0010	
0	0- 122	0001	– –| Positive portion of interval
0	4089-4095	0001	– –| Negative portion of interval
0	2048-4088	0000	
1	2048-4088	1111	
1	4089-4095	1110	– –| Negative portion of interval
1	0- 122	1110	– –| Positive portion of interval
1	123- 201	1101	
1	202- 260	1100	
1	261- 309	1011	
1	310- 355	1010	
1	356- 404	1001	
1	405-2047	1000	

TABLE 11/G.727

40 kbit/s quantizer decision levels and 5-bit outputs

DS	DLN	I	
		12345	
0	439-2047	01111	
0	405- 438	01110	
0	380- 404	01101	
0	356- 379	01100	
0	333- 355	01011	
0	310- 332	01010	
0	286- 309	01001	
0	261- 285	01000	
0	233- 260	00111	
0	202- 232	00110	
0	166- 201	00101	
0	123- 165	00100	
0	69- 122	00011	
0	0- 68	00010	– – Positive portion of interval
0	4089-4095	00010	– – Negative portion of interval
0	3961-4088	00001	
0	2048-3960	00000	
1	2048-3960	11111	
1	3961-4088	11110	
1	4089-4095	11101	– – Negative portion of interval
1	0- 68	11101	– – Positive portion of interval
1	69- 122	11100	
1	123- 165	11011	
1	166- 201	11010	
1	202- 232	11001	
1	233- 260	11000	
1	261- 285	10111	
1	286- 309	10110	
1	310- 332	10101	
1	333- 355	10100	
1	356- 379	10011	
1	380- 404	10010	
1	405- 438	10001	
1	439-2047	10000	

Note – The I values are transmitted starting with bit 1.

SUBTB

Inputs: DL (DLX in decoder), Y

Output: DLN (DLNX in decoder)

Function: Scale logarithmic version of difference signal by subtracting scale factor.

$DLN = (DL + 4096 - (Y >> 2)) \ \& \ 4095$

6.2.3 *Bit masking*

FIGURE 6/G.727

Bit masking

Input: I(k) or I´(k)

Outputs: $I_c(k)$

Function: Masking of quantized difference signal to extract the core bits.

Note: Figure 6/G.727 and equations are given for the encoder. They are also valid when substituting I´(k) for I(k) for the decoder.

$I_c = I \ >> E$, E = Enhancement Bits.

6.2.4 *Inverse adaptive quantizer*

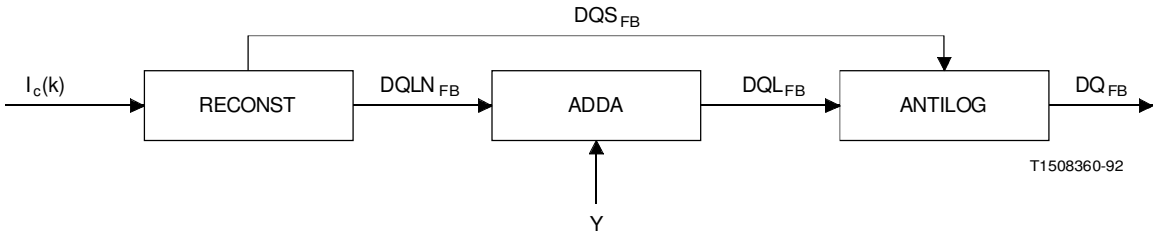

FIGURE 7/G.727

Inverse adaptive quantizer

RECONST

Input: I_c or I´(k)

Outputs: DQS_{FB} or DQS_{FF}, $DQLN_{FB}$ or $DQLN_{FF}$

Function: Reconstruction of quantized difference signal in the logarithmic domain.

Note: Figure 7/G.727, equations and tables are given for the feed-back path. They are also valid when substituting I´(k) for $I_c(k)$, DQS_{FF} for DQS_{FB} and $DQLN_{FF}$ for $DQLN_{FB}$.

$DQS_{FB} = I_C >> (C - 1)$

$DQS_{FF} = I' >> (E + C - 1)$

TABLE 12/G.727

16 kbit/s quantizer output levels

I′ or I_c	DQS_{FF} or	$DQLN_{FF}$ or
12	DQS_{FB}	$DQLN_{FB}$
01	0	365
00	0	116
11	1	116
10	1	365

TABLE 13/G.727

24 kbit/s quantizer output levels

I′ or I_c	DQS_{FF} or	$DQLN_{FF}$ or
123	DQS_{FB}	$DQLN_{FB}$
011	0	395
010	0	307
001	0	199
000	0	4085
111	1	4085
110	1	199
101	1	307
100	1	395

TABLE 14/G.727

32 kbit/s quantizer output levels

I' or I_c 1234	DQS_{FF} or DQS_{FB}	$DQLN_{FF}$ or $DQLN_{FB}$
0111	0	428
0110	0	377
0101	0	332
0100	0	285
0011	0	232
0010	0	165
0001	0	68
0000	0	3961
1111	1	3961
1110	1	68
1101	1	165
1100	1	232
1011	1	285
1010	1	332
1001	1	377
1000	1	428

TABLE 15/G.727

40 kbit/s quantizer output levels

I′ 12345	DQS_{FF}	$DQLN_{FF}$
01111	0	456
01110	0	419
01101	0	391
01100	0	367
01011	0	344
01010	0	321
01001	0	298
01000	0	273
00111	0	246
00110	0	217
00101	0	184
00100	0	145
00011	0	97
00010	0	34
00001	0	4035
00000	0	3832
11111	1	3832
11110	1	4035
11101	1	34
11100	1	97
11011	1	145
11010	1	184
11001	1	217
11000	1	246
10111	1	273
10110	1	298
10101	1	321
10100	1	344
10011	1	367
10010	1	391
10001	1	419
10000	1	456

Note 1 – The I values are received starting with bit 1.

ADDA

Inputs: $DQLN_{FB}$ or $DQLN_{FF}$, Y

Output: DQL_{FB} or DQL_{FF}

Function: Addition of scale factor to logarithmic version of quantized difference signal.

Note: Subscripts are given for the feed-back path. Figure 7/G.727 and equation are also valid when substituting $DQLN_{FF}$ for $DQLN_{FB}$ and DQL_{FF} for DQL_{FB}.

$DQL_{FB} = (DQLN_{FB} + (Y >> 2)) \& 4095$

ANTILOG

Inputs: DQL_{FB}, DQS_{FB}

Output: DQ_{FB}

Function: Convert quantized difference signal from the logarithmic to the linear domain.

Note: Figure 7/G.727 and equations are given for the feed-back path. The equations are also valid when substituting DQL_{FF} for DQL_{FB}, DQS_{FF} for DQS_{FB} and DQ_{FF} for DQ_{FB}.

$DS = DQL_{FB} >> 11$
$DEX = (DQL_{FB} >> 7) \& 15$ | Extract 4-bit exponent

$DMN = DQL_{FB} \& 127$ | Extract 7-bit mantissa

$DQT = (1 << 7) + DMN$
$DQMAG = \begin{cases} (DQT << 7) >> (14 - DEX), & DS = 0 \\ 0, & DS = 1 \end{cases}$ | Convert mantissa to linear using approximation $2^x = 1 + x$

$DQ_{FB} = (DQS << 14) + DQMAG$ | Attach sign bit to signed magnitude word

6.2.5 *Quantizer scale factor adaptation*

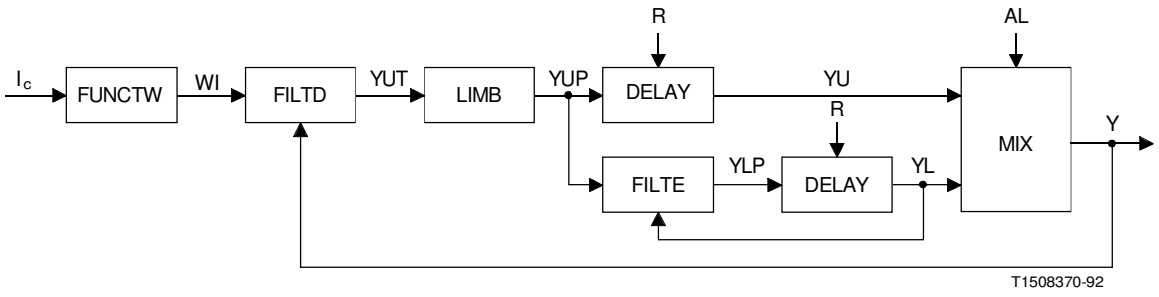

FIGURE 8/G.727

Quantizer scale factor adaptation

DELAY

Inputs: x, R (Optional)

Output: y

Function: Memory block. For the input x, the output is given by:

$$y(k) = \begin{cases} x(k-1), & R = 0 \\ \text{reset value given in column 4 of Table 3/G.727}, & R = 1 \end{cases} \quad \text{| reset}$$

FILTD

Inputs: WI, Y

Output: YUT

Function: Update of fast quantizer scale factor.

$DIF = ((WI << 5) + 131072 - Y)\ \&\ 131071$ | Compute difference

$DIFS = DIF >> 16$ |

$$DIFSX = \begin{cases} DIF >> 5, & DIFS = 0 \\ (DIF >> 5) + 4096, & DIFS = 1 \end{cases}$$ | Time constant is 1/32,
| Sign extension

$YUT = (Y + DIFSX)\ \&\ 8191$

FILTE

Inputs: YUP, YL

Output: YLP

Function: Update of slow quantizer scale factor.

$DIF = (YUP + ((1048576 - YL) >> 6))\ \&\ 16383$ | Compute difference

$DIFS = DIF >> 13$ | Time constant is 1/64

$$DIFSX = \begin{cases} DIF, & DIFS = 0 \\ DIF + 507904, & DIFS = 1 \end{cases}$$ |
| Sign extension
|

$YLP = (YL + DIFSX)\ \&\ 524287$

FUNCTW

Input: I_c

Output: WI

Function: Map quantizer output into logarithmic version of scale factor multiplier.

$IS = I_c >> (C - 1)$, $C = 2, 3, 4$.

For C = 2:

$$IM = \begin{cases} I_c \ \& \ 1, & IS = 0 \\ (3 - I_c) \ \& \ 1, & IS = 1 \end{cases}$$

$$WI = \begin{cases} 439, & IM = 1 \\ 4074, & IM = 0 \end{cases}$$

| Scale factor multipliers

For C = 3:

$$IM = \begin{cases} I_c \ \& \ 3, & IS = 0 \\ (7 - I_c) \ \& \ 3, & IS = 1 \end{cases}$$

$$WI = \begin{cases} 582, & IM = 3 \\ 137, & IM = 2 \\ 30, & IM = 1 \\ 4092, & IM = 0 \end{cases}$$

| Scale factor multipliers

For C = 4:

$$IM = \begin{cases} I_c \ \& \ 7, & IS = 0 \\ (15 - I_c) \ \& \ 7, & IS = 1 \end{cases}$$

$$WI = \begin{cases} 1108, & IM = 7 \\ 340, & IM = 6 \\ 184, & IM = 5 \\ 98, & IM = 4 \\ 50, & IM = 3 \\ 27, & IM = 2 \\ 4, & IM = 1 \\ 4084, & IM = 0 \end{cases}$$

| Scale factor multipliers

LIMB

Input: YUT

Output: YUP

Function: Limit quantizer scale factor.

$GEUL = ((YUT + 11264)\ \&\ 16383) >> 13$
$GELL = ((YUT + 15840)\ \&\ 16383) >> 13$

$$YUP = \begin{cases} 544, & GELL = 1 \\ 5120, & GEUL = 0 \\ \text{YUT}, & \text{otherwise} \end{cases}$$

| Set lower limit to 1.06
| Set upper limit to 10.00

MIX

Inputs: AL, YU, YL

Output: Y

Function: Form linear combination of fast and slow quantizer scale factors.

$DIF = (YU + 16384 - (YL >> 6))\ \&\ 16383$
$DIFS = DIF >> 13$

| Compute difference
|

$$DIFM = \begin{cases} DIF, & DIFS = 0 \\ (16384 - DIF)\ \&\ 8191, & DIFS = 1 \end{cases}$$

| Compute magnitude
| of difference
|

$PRODM = (DIFM * AL) >> 6$

| Compute magnitude
| of product

$$PROD = \begin{cases} PRODM, & DIFS = 0 \\ (16384 - PRODM)\ \&\ 16383, & DIFS = 1 \end{cases}$$

|
| Convert magnitude to
| two's complement

$Y = ((YL >> 6) + PROD)\ \&\ 8191$

6.2.6 *Adaptation speed control*

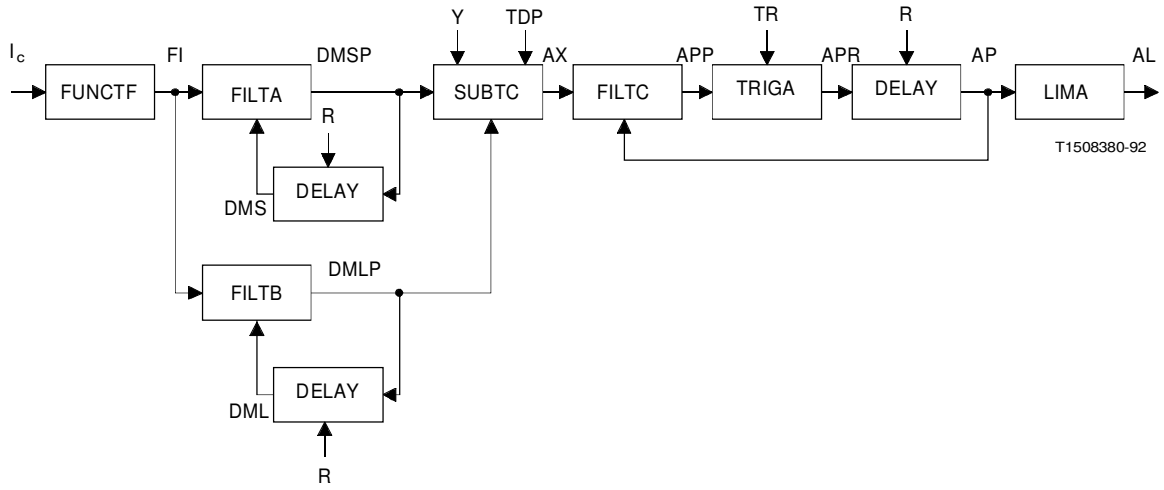

FIGURE 9/G.727

Adaptation speed control

DELAY

See § 6.2.5 for specification.

FILTA

Inputs: FI, DMS

Output: DMSP

Function: Update of short-term average of F(I).

$DIF = ((FI << 9) + 8192 - DMS) \,\&\, 8191$ | Compute difference
$DIFS = DIF >> 12$ |

$$DIFSX = \begin{cases} DIF >> 5, & DIFS = 0 \\ (DIF >> 5) + 3840, & DIFS = 1 \end{cases}$$

| Time constant is 1/32,
| Sign extension

$DMSP = (DIFSX + DMS) \,\&\, 4095$

FILTB

Inputs: FI, DML

Output: DMLP

Function: Update of long-term average of F(I).

$DIF = ((FI << 11) + 32768 - DML) \,\&\, 32767$ | Compute difference
$DIFS = DIF >> 14$ |

$$DIFSX = \begin{cases} DIF >> 7, & DIFS = 0 \\ (DIF >> 7) + 16128, & DIFS = 1 \end{cases}$$ | Time constant is 1/28, Sign extension

$DMLP = (DIFSX + DML) \,\&\, 16383$

FILTC

Inputs: AX, AP

Output: APP

Function: Low pass filter of speed control parameter.

$DIF = ((AX << 9) + 2048 - AP) \,\&\, 2047$ | Compute difference
$DIFS = DIF >> 10$ |

$$DIFSX = \begin{cases} DIF >> 4, & DIFS = 0 \\ (DIF >> 4) + 896, & DIFS = 1 \end{cases}$$ | Time constant is 1/16, Sign extension

$APP = (DIFSX + AP) \,\&\, 1023$

FUNCTF

Input: I_c

Output: FI

Function: Map quantizer output into the F(I) function.

$IS = I_c >> (C — 1)$, $C = 2, 3, 4$.

For C = 2:

$$IM = \begin{cases} I_c \ \& \ 1, & IS = 0 \\ (3 - I_c) \ \& \ 1, & IS = 1 \end{cases}$$

$$FI = \begin{cases} 7, & IM = 1 \\ 0, & IM = 0 \end{cases}$$

For C = 3:

$$IM = \begin{cases} I_c \ \& \ 3, & IS = 0 \\ (7 - I_c) \ \& \ 3, & IS = 1 \end{cases}$$

$$FI = \begin{cases} 7, & IM = 3 \\ 2, & IM = 2 \\ 1, & IM = 1 \\ 0, & IM = 0 \end{cases}$$

For C = 4:

$$IM = \begin{cases} I_c \ \& \ 7, & IS = 0 \\ (15 - I_c) \ \& \ 7, & IS = 1 \end{cases}$$

$$FI = \begin{cases} 0, & 0 \le IM \le 2 \\ 1, & 3 \le IM \le 5 \\ 3, & IM = 6 \\ 7, & IM = 7 \end{cases}$$

LIMA

Input: AP

Output: AL

Function: Limit speed control parameter.

$$AL = \begin{cases} 64, & AP \ge 256 \\ AP >> 2, & AP \le 255 \end{cases}$$

SUBTC

Inputs: DMSP, DMLP, TDP, Y

Output: AX

Function: Compute magnitude of the difference of short and long term functions of quantizer output sequence and then perform threshold comparison for quantizing speed control parameter.

$DIF = ((DMSP << 2) + 32768 - DMLP)\ \&\ 32767$ | Compute difference

$DIFS = DIF >> 14$ |

$$DIFM = \begin{cases} DIF, & DIFS = 0 \\ (32768 - DIF)\ \&\ 16383, & DIFS = 1 \end{cases}$$ | Compute magnitude of difference

$DTHR = DMLP >> 3$

$$AX = \begin{cases} 0, & Y \geq 1536 \text{ and } DIFM < DTHR \text{ and } TDP = 0 \\ 1, & \text{otherwise} \end{cases}$$

TRIGA

Inputs: TR, APP

Output: APR

Function: Speed control trigger block.

$$APR = \begin{cases} APP, & TR = 0 \\ 256, & TR = 1 \end{cases}$$

6.2.7 *Adaptive predictor and reconstructed signal calculator (feedback path)*

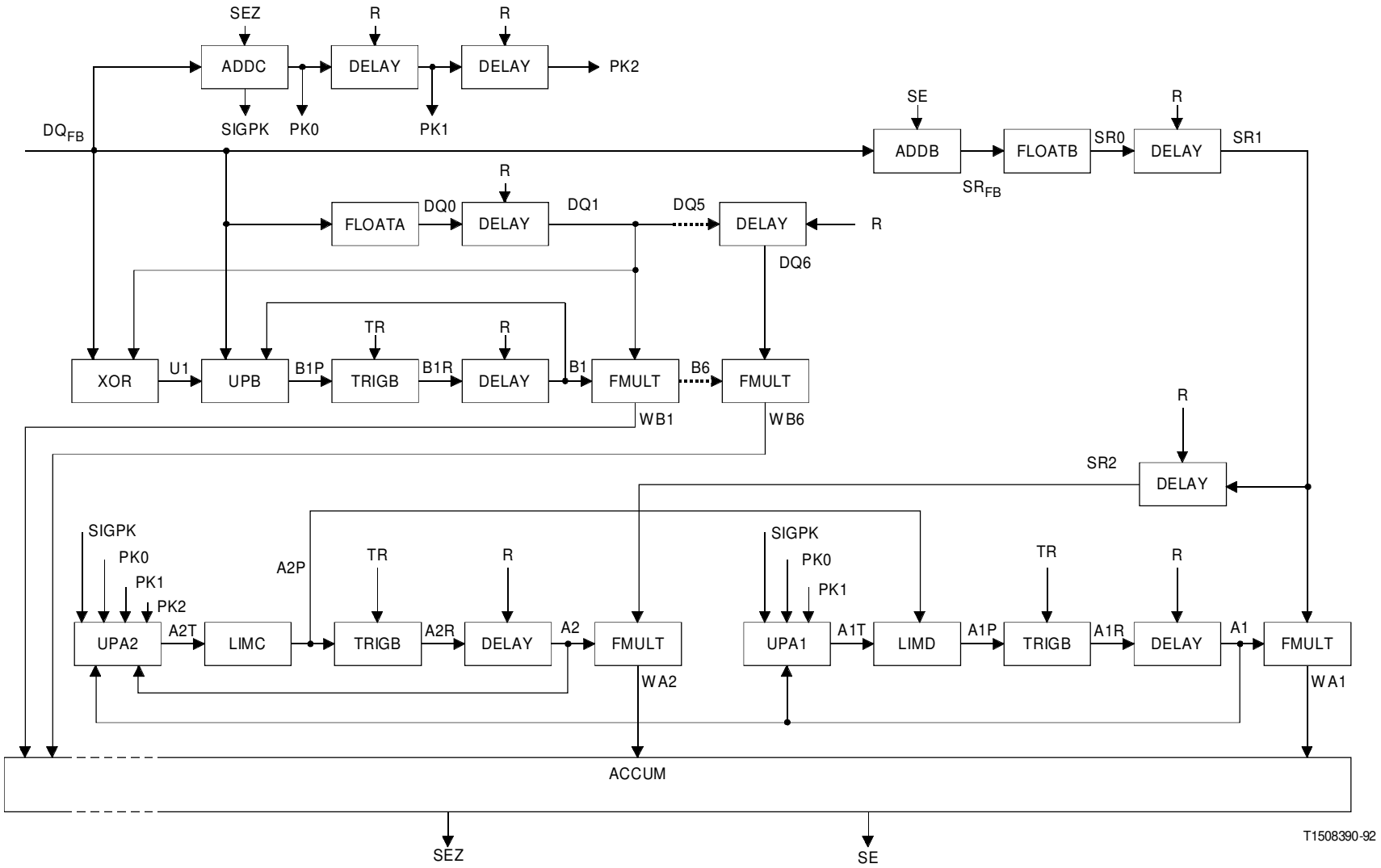

FIGURE 10/G.727

Adaptive predictor and reconstructed signal calculator

ACCUM

Inputs: WA1, WA2, WB1, WB2, WB3, WB4, WB5, WB6

Outputs: SE, SEZ

Function: Addition of predictor outputs to form the partial signal estimate (from the sixth order predictor) and the signal estimate.

$$SEZI = (((((((((WB1 + WB2) \,\&\, 65535) + WB3) \,\&\, 65535) + WB4) \,\&\, 65535) + WB5) \,\&\, 65535) + WB6) \,\&\, 65535$$

| Sum for partial signal estimate

$$SEI = (((SEZI + WA2) \,\&\, 65535) + WA1) \,\&\, 65535$$

| Complete sum for signal estimate

$$SEZ = SEZI >> 1$$

$$SE = SEI >> 1$$

ADDB

Inputs: DQ_{FB} or $DQ_{FF,}$ SE

Output: $SR_{FB\ or}\ DQ_{FF}$

Function: Addition of quantized difference signal and signal estimate to form reconstructed signal.

Note: Subscripts are given for the feed-back path. Equation is also valid when substituting DQ_{FF} for DQ_{FB} and DQ_{FF} for $DQ_{FB.}$

$$DQS_{FB} = DQ_{FB} >> 14$$

$$DQI = \begin{cases} DQ_{FB}, & DQS_{FB} = 0 \\ (65536 - (DQ_{FB} \,\&\, 16383)) \,\&\, 65535, & DQS_{FB} = 1 \end{cases}$$

| Convert signed magnitude to two's complement

$$SES = SE >> 14$$

$$SEI = \begin{cases} SE, & SES = 0 \\ (1 << 15) + SE, & SES = 1 \end{cases}$$

| Sign extension

$$SR_{FB} = (DQI + SEI) \,\&\, 65535$$

ADDC

Inputs: DQ_{FB}, SEZ

Output: PK0, SIGPK

Function: Obtain sign of addition of quantized difference signal and partial signal estimate.

$DQS_{FB} = DQ_{FB} >> 14$

$$DQI = \begin{cases} DQ_{FB}, & DQS_{FB} = 0 \\ (65536 - (DQ_{FB} \,\&\, 16383)) \,\&\, 65535, & DQS_{FB} = 1: \end{cases}$$

| Convert signed magnitude to two's complement

$SEZS = SEZ >> 14$

$$SEZI = \begin{cases} SEZ, & SEZS = 0 \\ (1 << 15) + SEZ, & SEZS = 1 \end{cases}$$

| Sign extension

$DQSEZ = (DQI + SEZI) \,\&\, 65535$

$PK0 = DQSEZ >> 15$

$$SIGPK = \begin{cases} 1, & DQSEZ = 0 \\ 0, & \text{otherwise} \end{cases}$$

DELAY

See § 6.2.5 for specification.

FLOATA

Input: DQ_{FB}

Output: DQ0

Function: Convert 15-bit signed magnitude to floating point.

$DQS_{FB} = DQ_{FB} >> 14$

$MAG = DQ_{FB} \,\&\, 16383$ | Compute magnitude

$$EXP = \begin{cases} 14, & 8192 \le MAG \\ 13, & 4096 \le MAG \le 8191 \\ . & . \\ . & . \\ . & . \\ 2, & 2 \le MAG \le 3 \\ 1, & MAG = 1 \\ 0, & MAG = 0 \end{cases}$$

| Compute exponent

$$MANT = \begin{cases} 1 << 5, & MAG = 0 \\ (MAG << 6) >> EXP, & \text{otherwise} \end{cases}$$

| Compute mantissa with a 1 in the most significant bit

$DQ0 = (DQS_{FB} << 10) + (EXP << 6) + MANT$ | Combine sign bit, 4 exponent bits and 6 mantissa bits into one 11-bit word

FLOATB

Input: SR_{FB}

Output: SR0

Function: Convert 16-bit two's complement to floating point.

$SRS = SR_{FB} >> 15$

$$MAG = \begin{cases} SR_{FB}, & SRS = 0 \\ (65536 - SR_{FB}) \ \& \ 32767, & SRS = 1 \end{cases}$$

| Compute magnitude

$$EXP = \begin{cases} 15, & 16384 \le MAG \\ 14, & 8192 \le MAG \le 16383 \\ . & . \\ . & . \\ . & . \\ 2, & 2 \le MAG \le 3 \\ 1, & MAG = 1 \\ 0, & MAG = 0 \end{cases}$$

| Compute exponent

$$MANT = \begin{cases} 1 << 5, & MAG = 0 \\ (MAG << 6) >> EXP, & \text{otherwise} \end{cases}$$

| Compute mantissa with a
| 1 in the most
| significant bit

$SR0 = (SRS << 10) + (EXP << 6) + MANT$

| combine sign bit, 4 exponent
| bits and 6 mantissa bits
| into one 11-bit word

FMULT

Inputs: An or Bn, SRn or DQn

Output: WAn or WBn

Note: Equations are given for An, SRn and WAn. The equations are also valid when substituting Bn for An, DQn for SRn and WBn for WAn.

Function: Multiply predictor coefficients with corresponding quantized difference signal or reconstructed signal. Multiplication is done in floating point format.

$AnS = An >> 15$

$$AnMAG = \begin{cases} An >> 2 & AnS = 0 \\ (16384 - (An >> 2)) \;\&\; 8191 & AnS = 1 \end{cases}$$

| Convert two's complement to signed magnitude

$$AnEXP = \begin{cases} 13, & 4096 \le AnMAG \\ 12, & 2048 \le AnMAG \le 4095 \\ . & . \\ . & . \\ . & . \\ 2, & 2 \le AnMAG \le 3 \\ 1, & AnMAG = 1 \\ 0, & AnMAG = 0 \end{cases}$$

| Compute exponent

$$AnMANT = \begin{cases} 1 << 5, & AnMAG = 0 \\ (AnMAG << 6) >> AnEXP, & \text{otherwise} \end{cases}$$

| Compute mantissa with a 1 in the most significant bit

$SRnS = SRn >> 10$
$SRnEXP = (SRn >> 6) \;\&\; 15$
$SRnMANT = SRn \;\&\; 63$

| Split floating point word into sign bit, exponent and mantissa

$WAnS = SRnS \ast\ast AnS$
$WAnEXP = SRnEXP + AnEXP$
$WAnMANT = ((SRnMANT * AnMANT) + 48) >> 4$

| Perform floating point multiplication

$$WAnMAG = \begin{cases} (WAnMANT << 7) >> (26 - WAnEXP), & WAnEXP \le 26 \\ ((WAnMANT << 7) << (WAnEXP - 26)) \;\&\; 32767, & WAnEXP > 26 \end{cases}$$

| Convert floating point to magnitude

$$WAn = \begin{cases} WAnMAG, & WAnS = 0 \\ (65536 - WAnMAG) \;\&\; 65535, & WAnS = 1 \end{cases}$$

| Convert mag. to two's complement

LIMC

Input: A2T

Output: A2P

Function: Limits on a_2 coefficient of second order predictor.

$A2UL = 12288$ | Upper limit of +0.75

$A2LL = 53248$ | Lower limit of −0.75

$$A2P = \begin{cases} A2LL, & 32768 \le A2T \le A2LL \\ A2UL, & A2UL \le A2T \le 32767 \\ A2T, & \text{otherwise} \end{cases}$$

LIMD

Inputs: A1T, A2P

Output: A1P

Function: Limits on a_1 coefficient of second order predictor.

$OME = 15360$ | (1 − epsilon) where
| epsilon = 1/16

$A1UL = (OME + 65536 - A2P)\ \&\ 65535$ | Compute upper limit

$A1LL = (A2P + 65536 - OME)\ \&\ 65535$ | Compute lower limit

$$A1P = \begin{cases} A1LL, & 32768 \le A1T \text{ and } A1T \le A1LL \\ A1UL, & A1UL \le A1T \text{ and } A1T \le 32767 \\ A1T, & \text{otherwise} \end{cases}$$

TRIGB

Inputs: TR, AnP or BnP or TDP

Output: AnR or BnR or TDR

Note: Equation is given for AnP and AnR. Equation is also valid when substituting BnP and BnR or TDP and TDR for AnP and AnR respectively.

Function: Predictor trigger block

$$AnR = \begin{cases} AnP, & TR = 0 \\ 0, & TR = 1 \end{cases}$$

UPA1

Inputs: PK0, PK1, A1, SIGPK

Output: A1T

Function: Update a_1 coefficient of second order predictor.

$PKS = PK0 \ast\ast PK1$ | 1-bit "exclusive or"

$$\mathrm{UGA1} = \begin{cases} 192, & PKS = 0 \text{ and } SIGPK = 0 \\ 65344, & PKS = 1 \text{ and } SIGPK = 0 \\ 0, & SIGPK = 1 \end{cases}$$
| Gain = ± 3/256

$A1S = A1 >> 15$

$$ULA1 = \begin{cases} (65536 - (\mathrm{A1} >> 8)) \ \& \ 65535, & A1S = 0 \\ (65536 - ((A1 >> 8) + 65280)) \ \& \ 65535, & A1S = 1 \end{cases}$$
| Leak factor = 1/256

$UA1 = (UGA1 + ULA1) \ \& \ 65535$ | Compute update

$A1T = (A1 + UA1) \ \& \ 65535$

UPA2

Inputs: PK0, PK1, PK2, A1, A2, SIGPK

Output: A2T

Function: Update a_2 coefficient of second order predictor.

$PKS1 = PK0 \ast\ast PK1$ | 1-bit "exclusive or"

$PKS2, = PK0 \ast\ast PK2$ | 1-bit "exclusive or"

$$UGA2A = \begin{cases} 16384, & PKS2 = 0 \\ 114688, & PKS2 = 1 \end{cases}$$

$A1S = A1 >> 15$

If $A1S = 0$ then

$$FA1 = \begin{cases} A1 << 2, & A1 \le 8191 \\ 8191 << 2, & A1 \ge 8192 \end{cases}$$

| Implement $f(a_1)$
| with limiting
| at +1/2

If $A1S = 1$ then

$$FA1 = \begin{cases} (A1 << 2) \,\&\, 131071, & A1 \ge 57345 \\ 24577 << 2, & A1 \le 57344 \end{cases}$$

| Implement $f(a_1)$
| with limiting
| at −1/2

$$FA = \begin{cases} FA1, & PKS1 = 1 \\ (131072 - FA1) \,\&\, 131071, & PKS1 = 0 \end{cases}$$

| Attach sign to
| result of $f(a_1)$
|

$UGA2B = (UGA2A + FA) \,\&\, 131071$ |
$UGA2S = UGA2B >> 16$ |

$$UGA2 = \begin{cases} UGA2B >> 7, & UGA2S = 0 \text{ and } SIGPK = 0 \\ (UGA2B >> 7) + 64512, & UGA2S = 1 \text{ and } SIGPK = 0 \\ 0, & SIGPK = 1 \end{cases}$$

| Gain calculation,
| gain = ± 1/128
|

$A2S = A2 >> 15$

$$ULA2 = \begin{cases} (65536 - (A2 >> 7)) \,\&\, 65535, & A2S = 0 \\ (65536 - ((A2 >> 7) + 65024)) \,\&\, 65535, & A2S = 1 \end{cases}$$

| Leak factor is
| 1/128
|

$UA2 = (UGA2 + ULA2) \,\&\, 65535$ | Compute update
$A2T = (A2 + UA2) \,\&\, 65535$ |

UPB

Inputs: Un, Bn, DQ_{FB}

Output: BnP

Function: Update for coefficients of sixth order predictor.

$DQMAG = DQ_{FB}$ & 16383

$$UGBn = \begin{cases} 128, & Un = 0 \text{ and } DQMAG \neq 0 \\ 65408, & Un = 1 \text{ and } DQMAG \neq 0 \\ 0, & DQMAG = 0 \end{cases}$$

| Gain = ± 1/128 or 0

$BnS = Bn >> 15$

$$ULBn = \begin{cases} (65536 - (Bn >> 8)) \,\&\, 65535, & BnS = 0 \\ (65536 - ((Bn >> 8) + 65280)) \,\&\, 65535 & BnS = 1 \end{cases}$$

| Leak factor = 1/256

$UBn = (UGBn + ULBn)$ & 65535
$BnP = (Bn + UBn)$ & 65535

| Compute update

XOR

Inputs: DQn, DQ_{FB}

Output: Un

Function: One bit “exclusive or” of sign of difference signal and sign of delayed difference signal.

$DQS_{FB} = DQ_{FB} >> 14$
$DQnS = DQn >> 10$

$Un = DQS_{FB} ** DQnS$ | 1-bit “exclusive or”

6.2.8 *Reconstructed signal calculator (feed-forward path)*

See § 6.2.7 for specification.

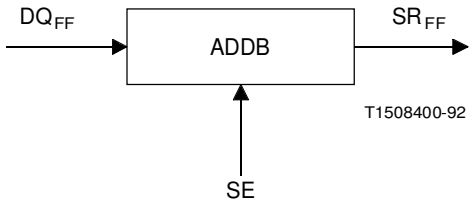

FIGURE 11/G.727

Reconstructed signal calculator (feed-forward path)

6.2.9 *Tone and transition detector*

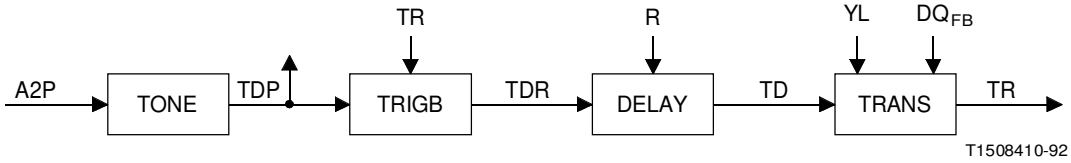

FIGURE 12/G.727

Tone and transition detector

DELAY

See § 6.2.5 for specification.

TONE

Input: A2P

Output: TDP

Function: Partial band signal detection

$$TDP = \begin{cases} 1, & 32768 \le A2P \text{ and } A2P < 53760 \\ 0, & \text{otherwise} \end{cases}$$

TRANS

Inputs: TD, YL, DQ_{FB}

Output: TR

Function: Transition detector.

$DQMAG = DQ_{FB}$ & 16383

$YLINT = YL >> 15$

$YLFRAC = (YL >> 10)$ & 31

$THR1 = (32 + YLFRAC) << YLINT$

$$THR2 = \begin{cases} 31 << 9, & YLINT > 8 \\ THR1, & \text{otherwise} \end{cases}$$

$DQTHR = (THR2 + (THR2 >> 1)) >> 1$

$$TR = \begin{cases} 1, & DQMAG > DQTHR \text{ and } TD = 1 \\ 0, & \text{otherwise} \end{cases}$$

TRIGB

See § 6.2.7 for specification.

6.2.10 *Output PCM format conversion and synchronous coding adjustment*

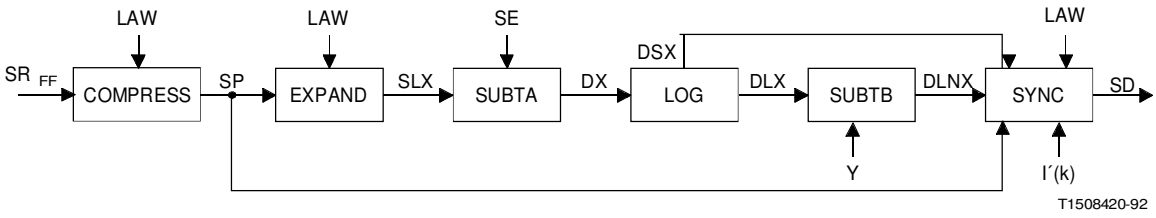

FIGURE 13/G.727

Output PCM format conversion and synchronous coding adjustment

COMPRESS (decoder only)

Input: SR_{FF}, LAW

Output: SP

Function: Convert from uniform PCM to either A-law or μ-law PCM.

$IS = SR_{FF} >> 15$

$$IM = \begin{cases} SR_{FF}, & IS = 0 \\ (65536 - SR_{FF}) \ \& \ 32767, & IS = 1 \end{cases}$$

| Convert two's
| complement to
| signed magnitude
|

$$IMAG = \begin{cases} IM, & LAW = 0 \\ IM >> 1 & LAW = 1 \text{ and } IS = 0 \\ (IM + 1) >> 1, & LAW = 1 \text{ and } IS = 1 \end{cases}$$

| μ-law
| A-law
|

then quantize IMAG (see note below) according to Recommendation G.711 using decision values (column 5 of Tables 1a/G.711, 1b/G.711, 2a/G.711 and 2b/G.711) in the following way:

$$SP = \begin{cases} \text{character signal after even bit inversion deduced from Table 1a/G.711 (column 6),} & IS = 0 \text{ and } LAW = 1 \\ \text{character signal after even bit inversion deduced from Table 1b/G.711 (column 6),} & IS = 1 \text{ and } LAW = 1 \\ \text{character signal of Table 2a/G.711 (column 6),} & IS = 0 \text{ and } LAW = 0 \\ \text{character signal of Table 2b/G.711 (column 6),} & IS = 1 \text{ and } LAW = 0 \end{cases}$$

Note – When IMAG is outside the range defined by the virtual decision level, SP must be set equal to the maximum PCM code word. For the purpose of clarification, examples of conversion for both A-law (after even bit inversion) and μ-law in the vicinity of the origin are given in the table below:

TABLE 16/G.727

Conversion for A-law and μ-law examples

IS	IMAG	PCM code word SP	
		A-law	μ-law
0	3	11010100	11111101
0	2	11010100	11111110
0	1	11010101	11111110
0	0	11010101	11111111
1	1	01010101	01111110
1	2	01010101	01111110
1	3	01010100	01111101

EXPAND

See § 6.2.1 for specification. Substitute SP for S as input and SLX for SL as output.

SUBTA

See § 6.2.1 for specification. Substitute SLX for SL as input and DX for D as output.

LOG

See § 6.2.2 for specification. Substitute DX for D as input, DLX for DL and DSX for DS as outputs.

SUBTB

See § 6.2.2 for specification. Substitute DLX for DL as input and DLNX for DLN as output.

SYNC (decoder only)

Inputs: I', SP, DLNX, DSX, LAW

Output: SD

Function: Re-encode output PCM sample in decoder for synchronous tandem coding.

$IS = I' >> (E + C - 1)$

For E + C = 2:

$$IM = \begin{cases} I' + 2, & IS = 0 \\ I' \ \& \ 1, & IS = 1 \end{cases}$$

ID is defined according to the following table:

TABLE 17/G.727

ID definition for E + C = 2

DSX	DLNX	ID	
0	261-2047	3	
0	0- 260	2	– Positive portion of decision interval
0	2048-4095	2	– Negative portion of decision interval
1	2048-4095	1	– Negative portion of decision interval
1	0- 260	1	– Positive portion of decision interval
1	261-2047	0	

For (E + C = 3):

$$IM = \begin{cases} I' + 4, & IS = 0 \\ I' \ \& \ 3, & IS = 1 \end{cases}$$

ID is defined according to the following table:

TABLE 18/G.727

ID definition for E + C = 3

DSX	DLNX	ID	
0	356-2047	7	
0	261- 355	6	
0	123- 260	5	
0	0- 122	4	– Positive portion of decision interval
0	2048-4095	4	– Negative portion of decision interval
1	2048-4095	3	– Negative portion of decision interval
1	0- 122	3	– Positive portion of decision interval
1	123- 260	2	
1	261- 355	1	
1	356-2047	0	

For (E + C = 4):

$$IM = \begin{cases} I' + 8, & IS = 0 \\ I' \ \& \ 7, & IS = 1 \end{cases}$$

ID is defined according to the following table:

TABLE 19/G.727

ID definition for E + C = 4

DSX	DLNX	ID	
0	405-2047	15	
0	356- 404	14	
0	310- 355	13	
0	261- 309	12	
0	202- 260	11	
0	123- 201	10	
0	0- 122	9	– –| Positive portion of decision interval
0	4089-4095	9	– –| Negative portion of decision interval
0	2048-4088	8	
1	2048-4088	7	
1	4089-4095	6	– –| Negative portion of decision interval
1	0- 122	6	– –| Positive portion of decision interval
1	123- 201	5	
1	202- 260	4	
1	261- 309	3	
1	310- 355	2	
1	356- 404	1	
1	405-2047	0	

For (E + C = 5):

$$IM = \begin{cases} I' + 16, & IS = 0 \\ I' \;\&\; 15, & IS = 1 \end{cases}$$

ID is defined according to the following table:

TABLE 20/G.727

ID definition for E + C = 5

DSX	DLNX	ID	
0	439-2047	31	
0	405- 438	30	
0	380- 404	29	
0	356- 379	28	
0	333- 355	27	
0	310- 332	26	
0	286- 309	25	
0	261- 285	24	
0	233- 260	23	
0	202- 232	22	
0	166- 201	21	
0	123- 165	20	
0	69- 122	19	
0	0- 68	18	– | Positive portion of decision interval
0	4089-4095	18	– | Negative portion of decision interval
0	3961-4088	17	
0	2048-3960	16	
1	2048-3960	15	
1	3961-4088	14	
1	4089-4095	13	– | Negative portion of decision interval
1	0- 68	13	– | Positive portion of decision interval
1	69- 122	12	
1	123- 165	11	
1	166- 201	10	
1	202- 232	9	
1	233- 260	8	
1	261- 285	7	
1	286- 309	6	
1	310- 332	5	
1	333- 355	4	
1	356- 379	3	
1	380- 404	2	
1	405- 438	1	
1	439-2047	0	

$$SD = \begin{cases} SP^{+}, & ID < IM \\ SP, & ID = IM \\ SP^{-}, & ID > IM \end{cases}$$

where

SP^{+} = the PCM code word that represents the next more positive PCM output level (when SP represents the most positive PCM output level, then SP^{+} is constrained to be SP).

SP^{-} = the PCM code word that represents the next more negative PCM output level (when SP represents the most negative PCM output level, then SP^{-} is constrained to be SP).

For the purposes of clarification, examples of re-encoding for both A-law (after even bit inversion) and μ-law in the vicinity of the origin are given in the table below:

TABLE 21/G.727

Re-encoding for A-law and μ-law

	A-law		μ-law	
Comparison of ID and IM	SP	SD	SP	SD
ID > IM	11010101	01010101	11111110	11111111
ID = IM	"	11010101	"	11111110
ID < IM	"	11010100	"	11111101
ID > IM	01010101	01010100	11111111	01111110
ID = IM	"	01010101	"	11111111
ID < IM	"	11010101	"	11111110
ID > IM	01010100	01010111	01111110	01111101
ID = IM	"	01010100	"	01111110
ID < IM	"	01010101	"	01111111

Note – SP (and SD) represent character signals defined according to Tables 1/G.711 and 2/G.711. See sub-block COMPRESS above for the exact representation of SP (and SD).

APPENDIX I

(to Recommendation G.727)

Digital test sequences for the verification of the algorithms in G.727

This Appendix gives information on the digital test sequences which have been chosen to verify implementations of the algorithms in G.727. Copies of the sequences on flexible diskettes together with a detailed description can be ordered from the ITU sales services (Please refer to collective letter No. 12/XV, 1991).

I.1 *Purpose of digital test sequences*

Digital sequences are used to verify the conformance of an implementation of a digital transcoding algorithm. The sequences are chosen to exercise the major arithmetic components and thus given a reasonable level of confidence of the compliance of an implementation with this Recommendation. Note that with a limited number of test sequences it is not possible to demonstrate 100% coverage of all states of the implementation. The more general issues involved in testing such algorithms are the subject of active research in the areas of VLSI testing and protocol conformance testing.

I.2 *Diskette interface and format*

Copies of the digital test sequences are available from the ITU on twelve 5-1/4"diskettes. The diskettes were created under MS-DOS operating system (version 3.2 or newer), and are of the 1.2 Mbyte high density double-sided 96 tracks per inch 51/4 MS-DOS format.