



INTERNATIONAL TELECOMMUNICATION UNION

# ITU-T

TELECOMMUNICATION  
STANDARDIZATION SECTOR  
OF ITU

# G.728

**Annex H**  
(05/99)

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

Digital transmission systems – Terminal equipments –  
Coding of analogue signals by methods other than PCM

---

Coding of speech at 16 kbit/s using low-delay code  
excited linear prediction

**Annex H: Variable bit rate LD-CELP operation  
mainly for DCME at rates less than 16 kbit/s**

ITU-T Recommendation G.728 – Annex H

(Previously CCITT Recommendation)

---

ITU-T G-SERIES RECOMMENDATIONS

**TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS**

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100–G.199
<b>INTERNATIONAL ANALOGUE CARRIER SYSTEM</b>	
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
<b>TESTING EQUIPMENTS</b>	
<b>TRANSMISSION MEDIA CHARACTERISTICS</b>	G.600–G.699
<b>DIGITAL TRANSMISSION SYSTEMS</b>	
TERMINAL EQUIPMENTS	G.700–G.799
General	G.700–G.709
Coding of analogue signals by pulse code modulation	G.710–G.719
<b>Coding of analogue signals by methods other than PCM</b>	<b>G.720–G.729</b>
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

*For further details, please refer to ITU-T List of Recommendations.*

**CODING OF SPEECH AT 16 kbit/s USING LOW-DELAY  
CODE EXCITED LINEAR PREDICTION**

**ANNEX H**

**Variable bit rate LD-CELP operation mainly  
for DCME at rates less than 16 kbit/s**

**Summary**

This annex contains the modifications to Recommendation G.728 LD-CELP speech coding algorithm needed to reduce the coding bit rate down to 12.8 and 9.6 kbit/s. These modifications include the modifications to the shape and gain codebooks.

This edition provides the additional description on the calculation of GSTATE(1) and the additional values for codebook related arrays.

This annex includes electronic material containing low bit rate LD\_CELP implementation test vectors.

**Source**

ITU-T Recommendation G.728, Annex H was revised by ITU-T Study Group 16 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on 27 May 1999.

## FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. The 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 Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 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 term *recognized operating agency (ROA)* includes any individual, company, corporation or governmental organization that operates a public correspondence service. The terms *Administration*, *ROA* and *public correspondence* are defined in the *Constitution of the ITU (Geneva, 1992)*.

## INTELLECTUAL PROPERTY RIGHTS

The ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. The 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, the ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

© ITU 2000

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.

## CONTENTS

	<b>Page</b>
H.1 Introduction.....	1
H.2 Principles of operation .....	1
H.2.1 Procedure to reduce coding bit rate .....	1
H.2.2 Principle of 12.8 kbit/s operation .....	2
H.2.3 Principle of 9.6 kbit/s operation .....	2
H.3 Modifications for 12.8 kbit/s operation .....	2
H.3.1 Pseudo-code.....	2
H.3.2 Additional new gain table.....	6
H.3.3 Change of coder parameter.....	7
H.4 Modifications for 9.6 kbit/s operation .....	7
H.4.1 Pseudo-code.....	7
H.4.2 Additional new gain table.....	12
H.4.3 Change of coder parameter.....	12

Included electronic file:

- Test data for 9.6 kbit/s operation
- Test data for 12.8 kbit/s operation



## **Recommendation G.728**

### **CODING OF SPEECH AT 16 kbit/s USING LOW-DELAY CODE EXCITED LINEAR PREDICTION**

#### **ANNEX H<sup>1</sup>**

#### **Variable bit rate LD-CELP operation mainly for DCME at rates less than 16 kbit/s**

*(revised in 1999)*

### **H.1 Introduction**

This annex contains the modifications to Recommendation G.728 LD-CELP speech coding algorithm needed to reduce the coding bit rate down to 12.8 and 9.6 kbit/s. These modifications include the modifications to the shape and gain codebooks.

This annex assumes that the reader is already familiar with the specifications in Recommendation G.728. The G.728 algorithm will not be discussed here; only the modifications to Recommendation G.728 will be described.

This annex consists of four subclauses. Subclause H.2 describes the principle of the operation for bit rate reduction. Subclause H.3 describes the modifications required for 12.8 kbit/s operation. Subclause H.4 describes the modifications required for 9.6 kbit/s operation.

### **H.2 Principles of operation**

#### **H.2.1 Procedure to reduce coding bit rate**

Reducing the bit rate without significantly changing the coding algorithm can be accomplished by reducing the size of the codebook. In the main body of Recommendation G.728, the codebook index consists of 10 bits corresponding to 1024 codebook vectors. A 2-bit reduction out of this 10-bit codebook index reduces the coding bit rate from 16 kbit/s to 12.8 kbit/s; a 4-bit reduction reduces the coding bit rate to 9.6 kbit/s.

The codebook index (bit 9 to bit 0) is divided into two sections: 7 bits (bit 9 to bit 3) for the shape codebook and 3 bits (bit 2 to bit 0) for the gain codebook. The 7-bit shape codebook consists of 128 code vectors; the 3-bit gain codebook consists of 8 scalar values that are symmetric with respect to zero.

The number of vectors in the shape codebook can be reduced as follows. The probability of vectors in the shape codebook obtained from normal speech samples does not have a uniform distribution. The occurrence probability of the codebook index numbers from 65 to 128 is observed to be much higher than those from 1 to 64. By taking advantage of this non-uniform distribution, it is possible to restrict the number of codebook vectors without introducing very much degradation in speech quality. For example, bit 9 in the shape codebook index is a candidate for reduction.

---

<sup>1</sup> This annex is an optional addition to Recommendation G.728 and is not required in order to correctly implement the encoder and decoder. This annex is intended to enhance the performance of Recommendation G.728 in some specific applications, such as in those to be used with digital circuit multiplication equipment. It is left to the implementers to choose whether to use this annex.

This annex includes electronic material containing data for low bit rate LD\_CELP implementation testing.

Another way to reduce the size of the shape codebook is to redesign the optimized codebook for each reduced bit rate operation. However, this would require more memory locations and a significant modification in the implementation.

The number of gain values in the gain codebook can be reduced as follows. To reduce the number of bits used for the gain codebook, it is better to redesign the reduced gain codebook, optimizing it for the reduced bit operations, because the gain codebook will occupy only a small amount of memory. Each reduced size gain codebook should be optimized based on the probability distribution of the gain values before quantization.

Reduction in the number of codebook index bits for 12.8 kbit/s operation is described in H.2.2 and that for 9.6 kbit/s operation is described in H.2.3.

## **H.2.2 Principle of 12.8 kbit/s operation**

Two bits must be eliminated from the 10-bit codebook index to achieve 12.8 kbit/s operation. Bit 9 of the shape codebook index is not used and a reduced gain codebook with four values is selected. Eliminating bit 9 from the shape codebook index restricts the codebook index numbers to the range 65 to 128. The four values of the gain codebook and the related values are optimized for 12.8 kbit/s operation; they are shown in Table H.1 (for floating-point calculation) and in Table H.2 (for fixed-point calculation) in H.3.2.

## **H.2.3 Principle of 9.6 kbit/s operation**

Reducing the 10-bit codebook index by 4 bits is necessary to achieve 9.6 kbit/s operation. Bits 9, 8 and 5 of the shape codebook index are eliminated, and the gain codebook is reduced from 8 to 4 values.

Eliminating bits 9, 8 and 5 from the shape codebook index limits the codebook index numbers to the ranges 97 to 100, 105 to 108, 113 to 116 and 121 to 124. The four values of the gain codebook and the related values are optimized for 9.6 kbit/s operation; they are shown in Table H.4 (for floating-point calculation) and Table H.5 (for fixed-point calculation) in H.4.2.

## **H.3 Modifications for 12.8 kbit/s operation**

### **H.3.1 Pseudo-code**

Only the block execution sequences are shown; the low level details of parameter passing are not described.

#### **H.3.1.1 Blocks 17 and 18 – Error calculator and the best codebook index selector**

Both the floating-point and fixed-point pseudo-codes for Blocks 17 and 18 are given in this subclause. These codes have been modified for 12.8 kbit/s operation and should be substituted for the original Blocks 17 and 18 described in 5.11/G.728. The floating-point pseudo-code is presented first.

```

Initialize DISTM to the largest number representable in the hardware
N1=NG_128/2
For J=65,66,...,NCWD, do the following
    J1=(J-1)*IDIM
    COR=0.
    For K=1,2,...,IDIM, do the next line
        COR=COR+PN(K)*Y(J1+K)           | compute inner product  $P_j$ 

```



```

If COR > 0, then do the next 3 lines
  IDXG=N1
  If COR < GB_128(1)*Y2(J), do the next line
    IDXG=1 | Best positive gain found

If COR ≤ 0, then do the next 3 lines
  IDXG=NG_128
  If COR > GB_128(3)*Y2(J), do the next line
    IDXG=3 | Best negative gain found

D=-G2_128(IDXG)*COR+GSQ_128(IDXG)*Y2(J) | Compute distortion  $\hat{D}$ 

If D < DISTM, do the next 3 lines
  DISTM=D | Save the lowest distortion
  IG=IDXG | and the best codebook indices
  IS=J | so far.

Repeat the above indented section for the next J

IS1=IS-NCWD/2

ICHAN=(IS1-1)*NG_128+(IG-1) | Concatenate shape and gain
                             | codebook indices.

```

Transmit ICHAN through communication channel. For bit-serial transmission, the most significant bit of ICHAN should be transmitted first. If ICHAN is represented by the 8-bit word b7, b6, b5, b4, b3, b2, b1, b0, then the order of the transmitted bits shall be b7, followed by b6, b5, b4, b3, b2, b1, b0 (b7 is the most significant bit).

The fixed-point version of the same module is given here. This fixed-point pseudo-code replaces the original Block 17 fixed-point code in G.3.9/G.728.

```

DISTM=2147483647
For J=65,66,...,NCWD, do the following
  J1=(J-1)*IDIM
  AA0=0
  For K=1,2,...,IDIM, do the next 2 lines
    P=PN(K)*Y(J1+K) | compute inner product  $P_j$ 
    AA0=AA0+P | NLS for AA0 is 7+11=18
  If AA0 < 0, set AA0=-AA0 | take absolute value
  IDXG=1
  P=GB_128(1)*Y2(J) | NLS for P is 13+5=18

  If AA0 ≥ P, set IDXG=IDXG+1

  AA0=AA0 >> 14 | NLS for AA0=4
  If AA0 > 32767, set AA0=32767 | clip AA0; AA0 in saturation mode
  AA1=GSQ_128(IDXG)*Y2(J) | NLSGSQ_128=11, NLSY2=5, so NLSAA1=16
  P=G2_128(IDXG)*AA0 | NLSG2_128=12, NLSAA0=4, so NLSA1=16
  AA1=AA1-P
  If AA1 < DISTM, do the next 3 lines
    DISTM=AA1 | double precision DISTM
    IG=IDXG
    IS=J
  Repeat the above indented section for the next J

  AA0=0 | Now find the sign bit
  J1=(IS-1)*IDIM
  For K=1,2,...,IDIM, do the next 2 lines
    P=PN(K)*Y(J1+K) | compute inner product
    AA0=AA0+P

```

```

If AA0 ≤ 0, set IG=IG+2

IS1=NCWD
IS1=IS1 >> 1
IS1=IS-IS1

ICHAN=(IS1-1)*NG_128+(IG-1)

```

In the above code, we used the following four lines:

```

AA0=AA0 >> 14           | NLS for AA0=4
If AA0 > 32767, set AA0=32767 | clip AA0
AA1=GSQ_128(IDXG)*Y2(J)   | NLSGSQ_128=11, NLSY2=5, so NLSAA1=16
P=G2_128(IDXG)*AA0        | NLSG2_128=12, NLSAA0=4, so NLSP=16

```

In DSP chips which have a "clipping" function; these lines can be replaced with the following code to give exactly the same results.

```

AA0=AA0 << 2           | NLS for AA0=20
AA0=CLIP(AA0)          | AA0 is in saturation mode
AA0=AA0 >> 16           | take high word; NLS for AA0=4
AA1=GSQ_128(IDXG)*Y2(J) | NLSGSQ_128=11, NLSY2=5, so NLSAA1=16
P=G2_128(IDXG)*AA0      | NLSG2_128=12, NLSAA0=4, so NLSP=16

```

The CLIP function and saturation mode refer to the concept of not allowing AA0 to overflow when the << 2 operation is performed. Instead of overflow, AA0 is set to the maximum positive or negative number, depending on its original sign. In this case, AA0 is always positive. This alternative is DSP dependent and may require more than a 32-bit accumulator. The alternative in the main pseudo-code can always be implemented.

### H.3.1.2 Block 19 – Excitation VQ codebook and Block 21 – Gain scaling unit

Both the floating-point and fixed-point pseudo-codes for Blocks 19 and 21 are given in this subclause. These codes have been modified for the case of 12.8 kbit/s operation and should be substituted for the original Block 19 described in 5.12/G.728. This is the floating-point version of the pseudo-code for Block 19, the excitation VQ codebook.

```

NN=(IS-1)*IDIM
For K=1,2,...,IDIM, do the next line
  YN(K)=GQ_128(IG)*Y(NN+K)

```

The floating-point version of the pseudo-code for Block 21, the gain scaling unit is given below.

```

For K=1,2,...,IDIM, do the next line
  ET(K) =GAIN*YN(K)

```

The fixed-point version of the same module is given here. This fixed-point pseudo-code replaces the original Blocks 19 and 21 fixed-point code in G.3.10/G.728.

For the fixed-point pseudo-code, we combine Blocks 19 and 21 into a single module. Both Y and GQ\_128 have fixed Q formats, Q11 and Q13, respectively. The value of GAIN has associated with it NLSGAIN. To get the maximum accuracy, the product GQ\_128(IG)\*GAIN is normalized to 32 bits before rounding to the upper 16 bits is performed. Let NNGQ\_128(I) be (1 + the number of left shifts needed to normalize the Q13 GQ\_128(I)), NNGQ\_128(I)=3 for I=1,3 and NNGQ\_128(I)=2 for I=2,4. The pseudo-code can thus be written as follows:

```

AA0=GQ_128(IG)*GAIN      | AA0 has NNGQ_128(IG) leading zeros
AA0=AA0 << NNGQ_128(IG)  | left shift NNGQ_128(IG) bits to
                           | normalize AA0

TMP=RND(AA0)              | round to upper 16 bits and assign to TMP
NLSAA0=13+NLSGAIN         | Q format of the product GQ_128(IG)*GAIN

```

NLSTMP=NLSAA0+NNGQ_128(IG)-16	Q format of TMP, because AA0 left
	shift by NNGQ_128(IG) bits then round
	and take upper 16 bits
NN=(IS-1)*IDIM	normalize selected shape
Call VSCALE(Y(NN+1), IDIM, IDIM, 14, TEMP, NLS)	codevector to 16 bits;
	put in TEMP
For K=1,2,...,IDIM, do the next 2 lines	
AA0=TMP*TEMP(K)	TMP and TEMP both normalized to 16 bits,
ET(K)=RND(AA0)	so the product has 1 leading zero.
	Directly rounding to high work gives us
	a 15-bit ET array.
NLSET=NLSTMP+11+NLS-16	calculate the NLS for ET.

### H.3.1.3 Block 29 – Decoder excitation VQ codebook and Block 31 – Decoder gain scaling unit

Both the floating-point and fixed-point pseudo-codes for Blocks 29 and 31 are given in this subclause. These codes have been modified for 12.8 kbit/s operation and should be substituted for the original Block 29 described in 5.14/G.728. This is the floating-point version of the pseudo-code for Block 29, the decoder excitation VQ codebook.

This block first extracts the 2-bit gain codebook index IG and the 6-bit shape codebook index IS from the received 8-bit channel index. The remainder of the operation is exactly the same as for Block 19 of the encoder.

```

ITMP=integer part of (ICHAN/NG_128)
IG=ICHAN-ITMP*NG_128+1
ITMP=ITMP+NCWD/2

NN=ITMP*IDIM
For K=1,2,...,IDIM, do the next line
  YN(K)=GQ_128(IG)*Y(NN+K)

```

The operation of Block 31, the decoder gain scaling unit, is exactly the same as for Block 21 of the encoder.

The fixed-point version of the same module is given here.

For the fixed-point pseudo-code, we combine Blocks 29 and 31 into a single module. Both Y and GQ\_128 have fixed Q formats, Q11 and Q13, respectively. The value of GAIN has associated with it NLSGAIN. To get the maximum accuracy, the product GQ\_128(IG)\*GAIN is normalized to 32 bits before rounding to the upper 16 bits is performed. Let NNGQ\_128(I) be (1 + the number of left shifts needed to normalize the Q13 GQ\_128(I)), so NNGQ\_128(I)=3 for I=1,3 and NNGQ\_128(I)=2 for I=2,4. The pseudo-code can thus be written as follows:

IS=ICHAN >> 2	
IG=ICHAN-IS*NG_128+1	
IS1=NCWD	
IS1=IS1 >> 1	
IS=IS+IS1+1	
AA0=GQ_128(IG)*GAIN	AA0 has NNGQ_128(IG) leading zeros
AA0=AA0 << NNGQ_128(IG)	left shift NNGQ_128(IG) bits to
	normalize AA0
TMP=RND(AA0)	round to upper 16 bits and assign to
	TMP
NLSAA0=13+NLSGAIN	Q format of the product GQ_128(IG)*GAIN
NLSTMP=NLSAA0+NNGQ_128(IG)-16	Q format of TMP, because AA0 left
	shift by NNGQ_128(IG) bits then round
	and take upper 16 bits
NN=(IS-1)*IDIM	normalize selected

```

Call VSCALE(Y(NN+1), IDIM, IDIM, 14 ,TEMP, NLS) | shape codevector to
                                                    | 16 bits;
                                                    | put in TEMP

For K=1,2,...,IDIM, do the next 2 lines
  AA0=TMP*TEMP(K) | TMP and TEMP both normalized to 16 bits,
  ET(K)=RND(AA0) | so the product has 1 leading zero.
                  | Directly rounding to high word gives us
                  | a 15-bit ET array.

NLSET=NLSTMP+11+NLS-16 | calculate the NLS for ET.

```

### H.3.1.4 Blocks 96 and 97 – Log-gain correction terms adder and limiter at –32 dB

Both the floating-point and fixed-point pseudo-codes for Block 96 and 97 are given in this subclause. These codes have been modified for 12.8 kbit/s operation and should be substituted for the original Blocks 96 and 97 described in G.3.16/G.728.

The floating-point pseudo-codes are:

```

GSTATE(1) = LOGGAIN + GCBLG_128(IG) + SHAPELG(IS)
If GSTATE(1) < -32.0, set GSTATE(1) = -32.0

```

The fixed-point pseudo-codes are:

```

AA0=LOGGAIN << 7 | Align decimal points at the
AA0=AA0 + (GCBLG_128(IG) << 5) | boundary between the high and
AA0=AA0 + (SHAPELG(IS) << 5) | low words of the accumulator.
AA0=AA0 >> 7 | Right shift back to Q9 format
If AA0 < -16384, set AA0=-16384 | Check lower limit
GSTATE(1)=AA0 | Lower 16 bit word saved

```

### H.3.2 Additional new gain table

This subclause gives the values for the gain codebook for 12.8 kbit/s operation. The floating-point values are given first. See Table H.1.

**Table H.1/G.728 – Floating-point values of gain codebook-related arrays**

Array index	1	2	3	4
GQ_128	0.525824	1.562449	–0.525824	–1.562449
GB_128	0.869912	*	–0.869912	*
G2_128	1.051648	3.124898	–1.051648	–3.124898
GSQ_128	0.276491	2.441247	0.276491	2.441247
GCBLG_128	–5.5831919	3.8761170	–5.5831919	3.8761170

The fixed-point values are given next. See Table H.2.

**Table H.2/G.728 – Fixed-point values of gain codebook-related arrays**

Array index	1	2	3	4
GQ_128(Q13)	4 308	12 800	−4 308	−12 800
GB_128(Q13)	7 126	*	−7 126	*
G2_128(Q12)	4 308	12 800	−4 308	−12 800
GSQ_128(Q11)	566	5 000	566	5 000
NNGQ_128(Q0)	3	2	3	2
GCBLG_128(Q11)	−11 434	7 938	−11 434	7 938

### H.3.3 Change of coder parameter

This subclause contains the new parameter, NG\_128. This parameter has been changed against NG (value = 8) of the G.728 for 12.8 kbit/s operation. See Table H.3.

**Table H.3/G.728 – Basic coder parameters of LD-CELP**

Name	Value	Description
NG_128	4	Gain codebook size (number of gain levels)

## H.4 Modifications for 9.6 kbit/s operation

### H.4.1 Pseudo-code

Only the block execution sequences are shown; the low level detail of parameter passing is not described.

#### H.4.1.1 Blocks 17 and 18 – Error calculator and best codebook index selector

Both the floating-point and fixed-point pseudo-codes for Blocks 17 and 18 are given in this subclause. These codes have been modified for 9.6 kbit/s operation and should be substituted for the original Blocks 17 and 18 described in 5.11/G.728. The floating-point pseudo-code is presented first.

```

Initialize DISTM to the largest number representable in the hardware
N1=NG_96/2
For K=1,2,3,4, do the following
  For K1=97,98,99,100, do the following
    J=(K-1)*8+K1
    J1=(J-1)*IDIM
    COR=0.
    For K2=1,2,...,IDIM, do the next line
      COR=COR+PN(K2)*Y(J1+K2)          | compute inner product  $P_j$ 

  If COR > 0, then do the next 3 lines
    IDXG=N1
    If COR < GB_96(1)*Y2(J), do the next line
      IDXG=1                          | Best positive gain found

  If COR ≤ 0, then do the next 3 lines
    IDXG=NG_96
    If COR > GB_96(3)*Y2(J), do the next line
      IDXG=3                          | Best negative gain found

```

```

D=-G2_96(IDXG)*COR+GSQ_96(IDXG)*Y2(J) | Compute distortion  $\hat{D}$ 
If D < DISTM, do the next 3 lines
DISTM=D | Save the lowest distortion
IG=IDXG | and the best codebook
IS=J | indices so far

Repeat the above indented section for the next K1.
Repeat the above indented section for the next K.

IS1=IS-(NCWD/2+NCWD/4)
IS2= integer part of (IS1/8)
IS2=IS2*4
IS3=IS1-IS2*2
IS1=IS2+IS3
ICHAN=(IS1-1)*NG_96+(IG-1) | Concatenate shape and
                             | gain codebook indices.

```

Transmit ICHAN through communication channel. For bit-serial transmission, the most significant bit of ICHAN should be transmitted first. If ICHAN is represented by the 6-bit word b5, b4, b3, b2, b1, b0, then the order of the transmitted bits shall be b5, and then b4, b3, b2, b1, b0 (b5 is the most significant bit).

The fixed-point version of the same module is given here. This fixed-point pseudo-code replaces the original Block 17 fixed-point code in G.3.9/G.728.

```

DISTM=2147483647
For K=1,2,3,4, do the following
  For K1=97,98,99,100, do the following
    J=(K-1)*8+K1
    J1=(J-1)*IDIM
    AA0=0
    For K2=1,2,...,IDIM, do the next 2 lines
      P=PN(K2)*Y(J1+K2) | compute inner product  $P_j$ 
      AA0=AA0+P | NLS for AA0 is 7+11=18
    If AA0 < 0, set AA0=-AA0 | take absolute value
    IDXG=1
    P=GB_96(1)*Y2(J) | NLS for P is 13+5=18
    If AA0 ≥ P, set IDXG=IDXG+1
    AA0=AA0 >> 14 | NLS for AA0=4
    If AA0 > 32767, set AA0=32767 | clip AA0; AA0 in saturation mode
    AA1=GSQ_96(IDXG)*Y2(J) | NLSGSQ_96=11, NLSY2=5, so NLSAA1=16
    P=G2_96(IDXG)*AA0 | NLSG2_96=12, NLSAA0=4, so NLSP=16
    AA1=AA1-P

    If AA1 < DISTM, do the next 3 lines
      DISTM=AA1 | double precision DISTM
      IG=IDXG
      IS=J

  Repeat the above indented section for the next K1.
Repeat the above indented section for the next K.

AA0=0 | Now find the sign bit
J1=(IS-1)*IDIM
For K=1,2,..., IDIM, do the next 2 lines
  P=PN(K)*Y(J1+K) | compute inner product
  AA0=AA0+P

```

```

If AA0 ≤ 0, set IG=IG+2
IS2=NCWD
IS1=IS2 >> 1
IS2=IS2 >> 2
IS1=IS-(IS1+IS2)
IS2=IS1 >> 3
IS2=IS2 << 2
IS3=IS2 << 1
IS3=IS1-IS3
IS1=IS2+IS3

ICHAN=(IS1-1)*NG_96+(IG-1)

```

In the above code, we used the following four lines:

AA0=AA0 >> 14	NLS for AA0=4
If AA0 > 32767, set AA0=32767	clip AA0
AA1=GSQ_96(IDXG)*Y2(J)	NLSGSQ_96=11, NLSY2=5, so NLSAA1=16
P=G2_96(IDXG)*AA0	NLSG2_96=12, NLSAA0=4, so NLSP=16

In DSP chips which have a "clipping" function, these lines can be replaced with the following code to give exactly the same results:

AA0=AA0 << 2	NLS for AA0=20
AA0=CLIP(AA0)	AA0 is in saturation mode
AA0=AA0 >> 16	take high word; NLS for AA0=4
AA1=GSQ_96(IDXG)*Y2(J)	NLSGSQ_96=11, NLSY2=5, so NLSAA1=16
P=G2_96(IDXG)*AA0	NLSG2_96=12, NLSAA0=4, so NLSP=16

The CLIP function and saturation mode refer to the concept of not allowing AA0 to overflow when the << 2 operation is performed. Instead of overflow, AA0 is set to the maximum positive or negative number, depending on its original sign. In this case, AA0 is always positive. This alternative is DSP dependent and may require more than a 32-bit accumulator. The alternative in the main pseudo-code can always be implemented.

#### H.4.1.2 Block 19 – Excitation VQ codebook and Block 21 – Gain scaling unit

Both the floating-point and fixed-point pseudo-codes for Blocks 19 and 21 are given in this subclause. These codes have been modified for 9.6 kbit/s operation and should be substituted for the original Block 19 described in 5.12/G.728. This is the floating-point version of the pseudo-code for Block 19, the excitation VQ codebook. First, the floating-point version of pseudo-code for Block 19 is presented.

```

NN=(IS-1)*IDIM
For K=1,2,..., IDIM, do the next line
    YN(K)=GQ_96(IG)*Y(NN+K)

```

The floating-point version of pseudo-code for Block 21, the gain scaling unit is given below.

```

For K=1,2,..., IDIM, do the next line
    ET(K)=GAIN*YN(K)

```

The fixed-point version of the same module is given here. This fixed-point pseudo-code replaces the original Blocks 19 and 21 fixed-point code in G.3.10/G.728.

For the fixed-point pseudo-code, we combine both Blocks 19 and 21 into a single module. Both Y and GQ\_96 have fixed Q formats, Q11 and Q13, respectively. The value of GAIN has associated with it NLSGAIN. To get the maximum accuracy, the product GQ\_96(IG)\*GAIN is normalized to 32 bits before rounding to the upper 16 bits is performed. Let NNGQ\_96(I) be (1 + the number of left shifts needed to

normalize the Q13 GQ\_96(I)). NNGQ\_96(I)=3 for I=1,3 and NNGQ\_96(I)=2 for I=2,4. The pseudo-code can thus be written as follows:

AA0=GQ_96(IG)*GAIN	AA0 has NNGQ_96(IG) leading zeros
AA0=AA0 << NNGQ_96(IG)	left shift NNGQ_96(IG) bits to
	normalize AA0
TMP=RND(AA0)	round to upper 16 bits and assign to TMP
NLSAA0=13+NLSGAIN	Q format of the product GQ_96(IG)*GAIN
NLSTMP=NLSAA0+NNGQ_96(IG)-16	Q format of TMP, because AA0 left shift
	by NNGQ_96(IG) bits then round and take
	16 upper bits
NN=(IS-1)*IDIM	normalize selected shape
Call VSCALE(Y(NN+1), IDIM, IDIM, 14, TEMP, NLS)	codevector to 16 bits; put
	in TEMP
For K=1,2,..., IDIM, do the next 2 lines	
AA0=TMP*TEMP(K)	TMP and TEMP both normalized to 16 bits,
ET(K) =RND(AA0)	so the product has 1 leading zero.
	Directly rounding to high work gives us
	a 15-bit ET array.
NLSET=NLSTMP+11+NLS-16	calculate the NLS for ET.

#### H.4.1.3 Block 29 – Decoder excitation VQ codebook and Block 31 – Decoder gain scaling unit

Both the floating-point and fixed-point pseudo-codes for Blocks 29 and 31 are given in this subclause. These codes have been modified for 9.6 kbit/s operation and should be substituted for the original Block 29 described in 5.14/G.728. This is the floating-point version of the pseudo-code for Block 29, the excitation VQ codebook.

This block first extracts the 2-bit gain codebook index IG and the 4-bit shape codebook index IS from the received 6-bit channel index. The remainder of the operation is exactly the same as for Blocks 19 and 21 of encoder.

```

ITMP=integer part of (ICHAN/NG_96)
IG=ICHAN-ITMP*NG_96+1
ITMP1=integer part of (ITMP/4)
ITMP2=ITMP-ITMP1*4
ITMP1=ITMP1*8
ITMP=ITMP1+ITMP2
ITMP=ITMP+(NCWD/2+NCWD/4)

NN=ITMP*IDIM
For K=1,2,...,IDIM, do the next line
    YN(K)=GQ_96(IG)*Y(NN+K)

```

The operation of Block 31, the decoder gain scaling unit, is exactly the same as for Block 21 of the encoder.

The fixed-point version of the same module is given here.

For the fixed-point pseudo-code, we combine Blocks 29 and 31 into a single module. Both Y and GQ\_96 have fixed Q formats, Q11 and Q13, respectively. The value of GAIN has associated with it NLSGAIN. To get the maximum accuracy, the product GQ\_96(IG)\*GAIN is normalized to 32 bits before rounding to the upper 16 bits is performed. Let NNGQ\_96(I) be (1 + the number of left shifts needed to normalize the Q13 GQ\_96(I)), NNGQ\_96(I)=3 for I=1,3 and NNGQ\_96(I)=2 for I=2,4. The pseudo-code can thus be written as follows:

```

IS=ICHAN >> 2
IG=ICHAN-IS*NG_96+1
IS1=IS >> 2

```



```

IS2=IS-IS1*4
IS1=IS1 << 3
IS=IS1+IS2
IS2=NCWD
IS1=IS2 >> 1
IS2=IS2 >> 2
IS=IS+IS1+IS2+1

AA0=GQ_96(IG)*GAIN          | AA0 has NNGQ_96(IG) leading zeros
AA0=AA0 << NNGQ_96(IG)      | left shift NNGQ_96(IG) bits to normalize
                              | AA0
TMP=RND(AA0)                 | round to upper 16 bits and assign to TMP

NLSAA0=13+NLSGAIN            | Q format of the product GQ_96(IG)*GAIN
NLSTMP=NLSAA0+NNGQ_96(IG)-16 | Q format of TMP, because AA0 left shift
                              | by NNGQ_96(IG) bits then round and take
                              | upper 16 bits
NN=(IS-1)*IDIM              | normalize selected shape
Call VSCALE(Y(NN+1), IDIM, IDIM, 14 ,TMP, NLS) | codevector to 16 bits;
                                                | put in TEMP

For K=1,2,...,IDIM, do the next 2 lines
  AA0=TMP*TEMP(K)            | TMP and TEMP both normalized to 16 bits,
  ET(K)=RND(AA0)             | so the product has 1 leading zero.
                              | Directly rounding to high work gives us
                              | a 15-bit ET array.

NLSET=NLSTMP+11+NLS-16      | calculate the NLS for ET.

```

#### H.4.1.4 Block 96 and 97 – Log-gain correction terms adder and limiter at –32 dB

Both the floating-point and fixed-point pseudo-codes for Blocks 96 and 97 are given in this subclause. These codes have been modified for 9.6 kbit/s operation and should be substituted for the original Blocks 96 and 97 described in G.3.16/G.728.

The floating-point pseudo-codes are:

```

GSTATE(1) = LOGGAIN + GCBLG_96(IG) + SHAPELG(IS)
If GSTATE(1) < -32.0, set GSTATE(1) = -32.0

```

The fixed-point pseudo-codes are:

```

AA0=LOGGAIN << 7          | Align decimal points at the
AA0=AA0 + (GCBLG_96(IG) << 5) | boundary between the high and
AA0=AA0 + (SHAPELG(IS) << 5) | low words of the accumulator.

AA0=AA0 >> 7              | Right shift back to Q9 format

If AA0 < -16384, set AA0=-16384 | Check lower limit
GSTATE(1)=AA0              | Lower 16 bit word saved

```

### H.4.2 Additional new gain table

This subclause gives the values for the gain codebook for 9.6 kbit/s operation. The floating-point values are given first. See Table H.4.

**Table H.4/G.728 – Floating-point values of gain codebook-related arrays**

Array index	1	2	3	4
GQ_96	0.564657	1.937714	−0.564657	−1.937714
GB_96	1.007492	*	−1.007492	*
G2_96	1.129314	3.875428	−1.129314	−3.875428
GSQ_96	0.318838	3.754736	0.318838	3.754736
GCBLG_96	−4.9643057	5.7457935	−4.9643057	5.7457935

The fixed-point values are given next. See Table H.5.

**Table H.5/G.728 – Fixed-point values of the gain codebook-related arrays**

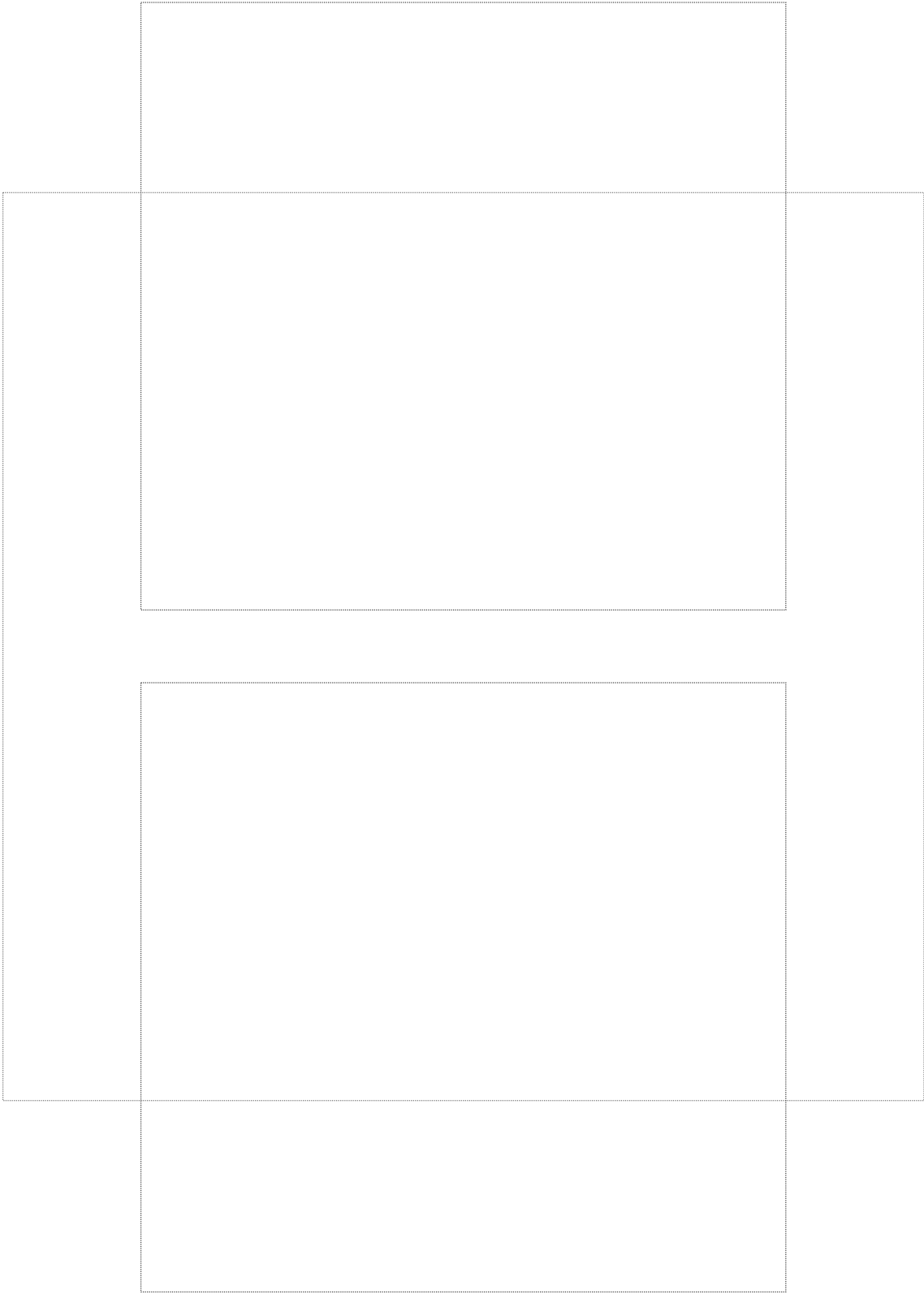
Array index	1	2	3	4
GQ_96(Q13)	4 626	15 874	−4 626	−15 874
GB_96(Q13)	8 253	*	−8 253	*
G2_96(Q12)	4 626	15 874	−4 626	−15 874
GSQ_96(Q11)	653	7 690	653	7 690
NNGQ_96(Q0)	3	2	3	2
GCBLG_96(Q11)	−10 167	11 767	−10 167	11 767

### H.4.3 Change of coder parameter

This subclause contains the new parameter, NG\_96. This parameter has been changed to NG (value = 8) of the G.728 for the case of 9.6 kbit/s operation. See Table H.6.

**Table H.6/G.728 – Basic coder parameters of LD-CELP**

Name	Value	Description
NG_96	4	Gain codebook size (number of gain values)



## ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
<b>Series G</b>	<b>Transmission systems and media, digital systems and networks</b>
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure and Internet protocol aspects
Series Z	Languages and general software aspects for telecommunication systems