

INTERNATIONAL TELECOMMUNICATION UNION

# ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

# H.263

(02/98)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Infrastructure of audiovisual services – Coding of moving video

---

## Video coding for low bit rate communication

ITU-T Recommendation H.263

(Previously CCITT Recommendation)

---



APPLE-1020

ITU-T H-SERIES RECOMMENDATIONS

**AUDIOVISUAL AND MULTIMEDIA SYSTEMS**

| | |
|---|---|
| Characteristics of transmission channels used for other than telephone purposes | H.10–H.19 |
| Use of telephone-type circuits for voice-frequency telegraphy | H.20–H.29 |
| Telephone circuits or cables used for various types of telegraph transmission or simultaneous transmission | H.30–H.39 |
| Telephone-type circuits used for facsimile telegraphy | H.40–H.49 |
| Characteristics of data signals | H.50–H.99 |
| CHARACTERISTICS OF VISUAL TELEPHONE SYSTEMS | H.100–H.199 |
| INFRASTRUCTURE OF AUDIOVISUAL SERVICES | |
| General | H.200–H.219 |
| Transmission multiplexing and synchronization | H.220–H.229 |
| Systems aspects | H.230–H.239 |
| Communication procedures | H.240–H.259 |
| **Coding of moving video** | **H.260–H.279** |
| Related systems aspects | H.280–H.299 |
| Systems and terminal equipment for audiovisual services | H.300–H.399 |

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

**ITU-T RECOMMENDATION H.263**

# VIDEO CODING FOR LOW BIT RATE COMMUNICATION


## Summary

This Recommendation specifies a coded representation that can be used for compressing the moving picture component of audio-visual services at low bit rates. The basic configuration of the video source coding algorithm is based on Recommendation H.261 and is a hybrid of inter-picture prediction to utilize temporal redundancy and transform coding of the remaining signal to reduce spatial redundancy. The source coder can operate on five standardized video source formats: sub-QCIF, QCIF, CIF, 4CIF and 16CIF, and can also operate using a broad range of custom video formats.

The decoder has motion compensation capability, allowing optional incorporation of this technique in the coder. Half pixel precision is used for the motion compensation, as opposed to Recommendation H.261 where full pixel precision and a loopfilter are used. Variable length coding is used for the symbols to be transmitted.

In addition to the basic video source coding algorithm, sixteen negotiable coding options are included for improved compression performance and the support of additional capabilities. Additional supplemental information may also be included in the bitstream for enhanced display capability and for external usage.



## Source

ITU-T Recommendation H.263 was revised by ITU-T Study Group 16 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on the 6th of February 1998.

## 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 expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

## 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 1998

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


| | | Page |
|---|---|---|
| 1 | Scope | 1 |
| 2 | References | 1 |
| 3 | Brief specification | 2 |
| 3.1 | Video input and output | 2 |
| 3.2 | Digital output and input | 2 |
| 3.3 | Sampling frequency | 2 |
| 3.4 | Source coding algorithm | 2 |
| 3.4.1 | Continuous Presence Multipoint and Video Multiplex mode | 3 |
| 3.4.2 | Unrestricted Motion Vector mode | 3 |
| 3.4.3 | Syntax-based Arithmetic Coding mode | 3 |
| 3.4.4 | Advanced Prediction mode | 3 |
| 3.4.5 | PB-frames mode | 3 |
| 3.4.6 | Forward Error Correction | 3 |
| 3.4.7 | Advanced INTRA Coding mode | 4 |
| 3.4.8 | Deblocking Filter mode | 4 |
| 3.4.9 | Slice Structured mode | 4 |
| 3.4.10 | Supplemental enhancement information | 4 |
| 3.4.11 | Improved PB-frames mode | 4 |
| 3.4.12 | Reference Picture Selection mode | 4 |
| 3.4.13 | Temporal, SNR and Spatial Scalability mode | 5 |
| 3.4.14 | Reference Picture Resampling mode | 5 |
| 3.4.15 | Reduced-Resolution Update mode | 5 |
| 3.4.16 | Independent Segment Decoding mode | 5 |
| 3.4.17 | Alternative INTER VLC mode | 5 |
| 3.4.18 | Modified Quantization mode | 5 |
| 3.5 | Bit rate | 6 |
| 3.6 | Buffering | 6 |
| 3.7 | Symmetry of transmission | 6 |
| 3.8 | Error handling | 6 |
| 3.9 | Multipoint operation | 7 |
| 4 | Source coder | 7 |
| 4.1 | Source format | 7 |
| 4.2 | Video source coding algorithm | 9 |
| 4.2.1 | GOBs, slices, macroblocks and blocks | 9 |
| 4.2.2 | Prediction | 11 |

| | | Page |
|---|---|---|
| 4.2.3 | Motion compensation | 12 |
| 4.2.4 | Quantization | 12 |
| 4.3 | Coding control | 13 |
| 4.4 | Forced updating | 13 |
| 4.5 | Byte alignment of start codes | 13 |
| 5 | Syntax and semantics | 13 |
| 5.1 | Picture layer | 22 |
| 5.1.1 | Picture Start Code (PSC) (22 bits) | 22 |
| 5.1.2 | Temporal Reference (TR) (8 bits) | 23 |
| 5.1.3 | Type Information (PTYPE) (Variable Length) | 23 |
| 5.1.4 | Plus PTYPE (PLUSPTYPE) (Variable Length) | 24 |
| 5.1.5 | Custom Picture Format (CPFMT) (23 bits) | 27 |
| 5.1.6 | Extended Pixel Aspect Ratio (EPAR) (16 bits) | 28 |
| 5.1.7 | Custom Picture Clock Frequency Code (CPCFC) (8 bits) | 28 |
| 5.1.8 | Extended Temporal Reference (ETR) (2 bits) | 28 |
| 5.1.9 | Unlimited Unrestricted Motion Vectors Indicator (UUI) (Variable length). | 29 |
| 5.1.10 | Slice Structured Submode bits (SSS) (2 bits) | 29 |
| 5.1.11 | Enhancement Layer Number (ELNUM) (4 bits) | 29 |
| 5.1.12 | Reference Layer Number (RLNUM) (4 bits) | 29 |
| 5.1.13 | Reference Picture Selection Mode Flags (RPSMF) (3 bits) | 29 |
| 5.1.14 | Temporal Reference for Prediction Indication (TRPI) (1 bit) | 30 |
| 5.1.15 | Temporal Reference for Prediction (TRP) (10 bits) | 30 |
| 5.1.16 | Back-Channel message Indication (BCI) (Variable length) | 30 |
| 5.1.17 | Back-Channel Message (BCM) (Variable length) | 30 |
| 5.1.18 | Reference Picture Resampling Parameters (RPRP) (Variable length) | 30 |
| 5.1.19 | Quantizer Information (PQUANT) (5 bits) | 30 |
| 5.1.20 | Continuous Presence Multipoint and Video Multiplex (CPM) (1 bit) | 31 |
| 5.1.21 | Picture Sub-Bitstream Indicator (PSBI) (2 bits) | 31 |
| 5.1.22 | Temporal Reference for B-pictures in PB-frames ($TR_B$) (3/5 bits) | 31 |
| 5.1.23 | Quantization information for B-pictures in PB-frames (DBQUANT) (2 bits) | 31 |
| 5.1.24 | Extra Insertion Information (PEI) (1 bit) | 31 |
| 5.1.25 | Supplemental Enhancement Information (PSUPP) (0/8/16 ... bits) | 31 |
| 5.1.26 | Stuffing (ESTUF) (Variable length) | 32 |
| 5.1.27 | End Of Sequence (EOS) (22 bits) | 32 |
| 5.1.28 | Stuffing (PSTUF) (Variable length) | 32 |
| 5.2 | Group of Blocks Layer | 32 |
| 5.2.1 | Stuffing (GSTUF) (Variable length) | 32 |

| | | Page |
|---|---|---|
| 5.2.2 | Group of Block Start Code (GBSC) (17 bits) | 33 |
| 5.2.3 | Group Number (GN) (5 bits) | 33 |
| 5.2.4 | GOB Sub-Bitstream Indicator (GSBI) (2 bits) | 33 |
| 5.2.5 | GOB Frame ID (GFID) (2 bits) | 33 |
| 5.2.6 | Quantizer Information (GQUANT) (5 bits) | 33 |
| 5.3 | Macroblock layer | 33 |
| 5.3.1 | Coded macroblock indication (COD) (1 bit) | 34 |
| 5.3.2 | Macroblock type & Coded Block Pattern for Chrominance (MCBPC) (Variable length) | 34 |
| 5.3.3 | Macroblock mode for B-blocks (MODB) (Variable length) | 37 |
| 5.3.4 | Coded Block Pattern for B-blocks (CBPB) (6 bits) | 37 |
| 5.3.5 | Coded Block Pattern for luminance (CBPY) (Variable length) | 37 |
| 5.3.6 | Quantizer Information (DQUANT) (2 bits/Variable Length) | 37 |
| 5.3.7 | Motion Vector Data (MVD) (Variable length) | 38 |
| 5.3.8 | Motion Vector Data ($MVD_{2-4}$) (Variable length) | 40 |
| 5.3.9 | Motion Vector Data for B-macroblock (MVDB) (Variable length) | 40 |
| 5.4 | Block layer | 40 |
| 5.4.1 | DC coefficient for INTRA blocks (INTRADC) (8 bits) | 41 |
| 5.4.2 | Transform Coefficient (TCOEF) (Variable length) | 41 |
| 6 | Decoding process | 44 |
| 6.1 | Motion compensation | 44 |
| 6.1.1 | Differential motion vectors | 44 |
| 6.1.2 | Interpolation for subpixel prediction | 46 |
| 6.2 | Coefficients decoding | 46 |
| 6.2.1 | Inverse quantization | 46 |
| 6.2.2 | Clipping of reconstruction levels | 46 |
| 6.2.3 | Zigzag positioning | 47 |
| 6.2.4 | Inverse transform | 47 |
| 6.3 | Reconstruction of blocks | 47 |
| 6.3.1 | Summation | 47 |
| 6.3.2 | Clipping | 48 |
| Annex A – Inverse transform accuracy specification | | 48 |
| Annex B – Hypothetical Reference Decoder | | 49 |

Page

Annex C – Considerations for multipoint ..... 51
C.1 Freeze picture request ..... 51
C.2 Fast update request ..... 51
C.3 Freeze picture release ..... 51
C.4 Continuous Presence Multipoint and Video Multiplexing (CPM) ..... 51
C.4.1 End Of Sub-Bitstream code (EOSBS) (23 bits) ..... 52
C.4.2 Ending Sub-Bitstream Indicator (ESBI) (2 bits) ..... 53
Annex D – Unrestricted Motion Vector mode ..... 53
D.1 Motion vectors over picture boundaries ..... 53
D.1.1 Restrictions for motion vector values ..... 54
D.2 Extension of the motion vector range ..... 54
Annex E – Syntax-based Arithmetic Coding mode ..... 56
E.1 Introduction ..... 56
E.2 Specification of SAC encoder ..... 57
E.3 Specification of SAC decoder ..... 58
E.4 Syntax ..... 58
E.5 PSC_FIFO ..... 59
E.6 Header layer symbols ..... 59
E.7 Macroblock and Block layer symbols ..... 60
E.8 SAC models ..... 61
Annex F – Advanced Prediction mode ..... 64
F.1 Introduction ..... 64
F.2 Four motion vectors per macroblock ..... 64
F.3 Overlapped motion compensation for luminance ..... 65
Annex G – PB-frames mode ..... 68
G.1 Introduction ..... 68
G.2 PB-frames and INTRA blocks ..... 69
G.3 Block layer ..... 69
G.4 Calculation of vectors for the B-picture in a PB-frame ..... 69
G.5 Prediction of a B-block in a PB-frame ..... 70
Annex H – Forward error correction for coded video signal ..... 72
H.1 Introduction ..... 72
H.2 Error correction framing ..... 72
H.3 Error correcting code ..... 72

Page

H.4 Relock time for error corrector framing ..... 72

Annex I – Advanced INTRA Coding mode ..... 73

I.1 Introduction ..... 73

I.2 Syntax ..... 74

I.3 Decoding process ..... 74

Annex J – Deblocking Filter mode ..... 80

J.1 Introduction ..... 80

J.2 Relation to UMV and AP modes (Annexes D and F) ..... 81

J.3 Definition of the deblocking edge filter ..... 81

Annex K – Slice Structured mode ..... 85

K.1 Introduction ..... 85

K.2 Structure of slice layer ..... 86

K.2.1 Stuffing (SSTUF) (Variable length) ..... 86

K.2.2 Slice Start Code (SSC) (17 bits) ..... 86

K.2.3 Slice Emulation Prevention Bit 1 (SEPB1) (1 bit) ..... 86

K.2.4 Slice Sub-Bitstream Indicator (SSBI) (4 bits) ..... 86

K.2.5 Macroblock Address (MBA) (5/6/7/9/11/12/13/14 bits) ..... 87

K.2.6 Slice Emulation Prevention Bit 2 (SEPB2) (1 bit) ..... 87

K.2.7 Quantizer Information (SQUANT) (5 bits) ..... 87

K.2.8 Slice Width Indication in Macroblocks (SWI) (3/4/5/6/7 bits) ..... 87

K.2.9 Slice Emulation Prevention Bit 3 (SEPB3) (1 bit) ..... 88

Annex L – Supplemental Enhancement Information Specification ..... 88

L.1 Introduction ..... 88

L.2 PSUPP format ..... 88

L.3 Do Nothing ..... 89

L.4 Full-Picture Freeze Request ..... 89

L.5 Partial-Picture Freeze Request ..... 89

L.6 Resizing Partial-Picture Freeze Request ..... 90

L.7 Partial-Picture Freeze-Release Request ..... 90

L.8 Full-Picture Snapshot Tag ..... 90

L.9 Partial-Picture Snapshot Tag ..... 91

L.10 Video Time Segment Start Tag ..... 91

L.11 Video Time Segment End Tag ..... 91

L.12 Progressive Refinement Segment Start Tag ..... 91

L.13 Progressive Refinement Segment End Tag ..... 91

Page

L.14 Chroma Keying Information ..... 92
L.15 Extended Function Type ..... 94

Annex M – Improved PB-frames mode ..... 94
M.1 Introduction ..... 94
M.2 $B_{PB}$-macroblock prediction modes ..... 95
M.2.1 Bidirectional prediction ..... 95
M.2.2 Forward prediction ..... 95
M.2.3 Backward prediction ..... 95
M.3 Calculation of vectors for bidirectional prediction of a the B-macroblock ..... 95
M.4 MODB table ..... 95

Annex N – Reference Picture Selection mode ..... 96
N.1 Introduction ..... 96
N.2 Video source coding algorithm ..... 97
N.3 Channel for back-channel messages ..... 97
N.3.1 Separate logical channel mode ..... 98
N.3.2 Videomux mode ..... 98
N.4 Syntax ..... 98
N.4.1 Forward channel ..... 98
N.4.2 Back-Channel Message (BCM) syntax ..... 100
N.5 Decoder process ..... 101

Annex O – Temporal, SNR, and Spatial Scalability mode ..... 102
O.1 Overview ..... 102
O.1.1 Temporal scalability ..... 102
O.1.2 SNR scalability ..... 103
O.1.3 Spatial scalability ..... 104
O.1.4 Multilayer scalability ..... 105
O.2 Transmission order of pictures ..... 106
O.3 Picture layer syntax ..... 108
O.4 Macroblock layer syntax ..... 108
O.4.1 Coded macroblock indication (COD) (1 bit) ..... 110
O.4.2 MBTYPE/MCBPC (VLC) ..... 110
O.4.3 Coded Block Pattern for Chrominance (CBPC) (Variable length) ..... 112
O.4.4 Coded Block Pattern for Luminance (CBPY) (Variable length) ..... 112
O.4.5 Quantizer Information (DQUANT) (2 bits/Variable length) ..... 113
O.4.6 Motion vector data (MVDFW, MVDBW) (Variable length) ..... 113

Page

O.5 Motion vector decoding ..... 113
O.5.1 Differential motion vectors ..... 113
O.5.2 Motion vectors in direct mode ..... 113
O.6 Interpolation filters ..... 113
Annex P – Reference picture resampling ..... 116
P.1 Introduction ..... 116
P.2 Syntax ..... 119
P.2.1 Warping Displacement Accuracy (WDA) (2 bits) ..... 119
P.2.2 Warping parameters (Variable length) ..... 119
P.2.3 Fill Mode (FILL_MODE) (2 bits) ..... 120
P.2.4 Fill Color Specification (Y_FILL, $C_B$_EPB, $C_B$_FILL, $C_R$_EPB, $C_R$_FILL) (26 bits) ..... 120
P.3 Resampling algorithm ..... 121
P.4 Example of implementation ..... 124
P.4.1 Displacements of virtual points ..... 124
P.4.2 Resampling algorithm ..... 124
P.5 Factor-of-4 resampling ..... 126
P.5.1 Factor-of-4 upsampling ..... 126
P.5.2 Factor-of-4 downsampling ..... 128
Annex Q – Reduced-Resolution Update mode ..... 129
Q.1 Introduction ..... 129
Q.2 Decoding procedure ..... 130
Q.2.1 Reference preparation ..... 131
Q.2.2 Macroblock layer decoding ..... 131
Q.2.3 Picture store ..... 132
Q.2.4 Display ..... 132
Q.3 Extension of referenced picture ..... 133
Q.4 Reconstruction of motion vectors ..... 134
Q.5 Enlarged overlapped motion compensation for luminance ..... 136
Q.6 Upsampling of the reduced-resolution reconstructed prediction error ..... 138
Q.6.1 Upsampling procedure for the pixels inside a 16 × 16 reconstructed prediction error block ..... 140
Q.6.2 Upsampling procdedure for the pixels at the boundary of 16 × 16 reconstructed prediction error block ..... 141
Q.7 Block boundary filter ..... 141
Q.7.1 Definition of the default block boundary filter ..... 142

Page

Q.7.2 Definition of the block boundary filter when Deblocking Filter mode is used ..... 142

Annex R – Independent Segment Decoding mode ..... 143
R.1 Introduction ..... 143
R.2 Mode operation ..... 143
R.3 Constraints on usage ..... 144
R.3.1 Constraint on segment shapes ..... 144
R.3.2 Constraint on changes of segment shapes ..... 144

Annex S – Alternative INTER VLC mode ..... 145
S.1 Introduction ..... 145
S.2 Alternative INTER VLC for coefficients ..... 145
S.2.1 Encoder action ..... 145
S.2.2 Decoder action ..... 145
S.3 Alternative INTER VLC for CBPY ..... 146

Annex T – Modified Quantization mode ..... 146
T.1 Introduction ..... 146
T.2 Modified DQUANT Update ..... 146
T.2.1 Small-step QUANT alteration ..... 146
T.2.2 Arbitrary QUANT selection ..... 147
T.3 Altered quantization step size for chrominance coefficients ..... 147
T.4 Modified coefficient range ..... 148
T.5 Usage restrictions ..... 148

Appendix I – Error tracking ..... 149
I.1 Introduction ..... 149
I.2 Error tracking ..... 149

Appendix II – Recommended Optional Enhancement ..... 150
II.1 Introduction ..... 150
II.2 Levels of preferred mode support ..... 151
II.2.1 Level 1 preferred modes ..... 151
II.2.2 Level 2 preferred modes ..... 152
II.2.3 Level 3 preferred modes ..... 152
II.3 Picture formats and picture clock frequencies ..... 153

**Recommendation H.263**

# VIDEO CODING FOR LOW BIT RATE COMMUNICATION

*(revised in 1998)*

## 1 Scope

This Recommendation specifies a coded representation that can be used for compressing the moving picture component of audio-visual services at low bit rates. The basic configuration of the video source coding algorithm is based on Recommendation H.261. Sixteen negotiable coding options are included for improved performance and increased functionality. This Recommendation contains Version 2 of Recommendation H.263, which is fully compatible with the original Recommendation, adding only optional features to the original Version 1 content of the Recommendation.

## 2 References

The following Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below.

[1] ITU-R Recommendation BT.601-5 (1995), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios.*

Reference [1] is referenced herein to define the ($Y$, $C_B$, $C_R$) colour space and its 8-bit integer representation for pictures used by video codecs designed according to this Recommendation. (Reference [1] is not used to define any other aspects of this Recommendation.)

The following additional ITU-T Recommendations are mentioned for illustration purposes in this text; however, they do not constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

[2] ITU-T Recommendation H.223 (1996), *Multiplexing protocol for low bit rate multimedia communication.*

[3] ITU-T Recommendation H.242 (1997), *System for establishing communication between audiovisual terminals using digital channels up to 2 Mbit/s.*

[4] ITU-T Recommendation H.245 (1998), *Control protocol for multimedia communication.*

[5] ITU-T Recommendation H.261 (1993), *Video codec for audiovisual services at $p \times 64$ kbit/s.*

[6] ITU-T Recommendation H.262 (1995) | ISO/IEC 13818-2:1996, *Information technology – Generic coding of moving pictures and associated audio information: Video.*

[7] ITU-T Recommendation H.324 (1998), *Terminal for low bit rate multimedia communication.*

## 3 Brief specification

An outline block diagram of the codec is given in Figure 1.

### 3.1 Video input and output

To permit a single Recommendation to cover use in and between regions using 625- and 525-line television standards, the standardized source formats on which the source coder operates are based on a Common Intermediate Format (CIF). It is also possible using external negotiation (for example, Recommendation H.245), to enable the use of a wide range of optional custom source formats. The standards of the input and output television signals, which may, for example, be composite or component, analogue or digital and the methods of performing any necessary conversion to and from the source coding format are not subject to recommendation.

### 3.2 Digital output and input

The video coder provides a self-contained digital bitstream which may be combined with other multi-facility signals (for example as defined in Recommendation H.223). The video decoder performs the reverse process.

### 3.3 Sampling frequency

Pictures are sampled at an integer multiple of the video line rate. This sampling clock and the digital network clock are asynchronous.

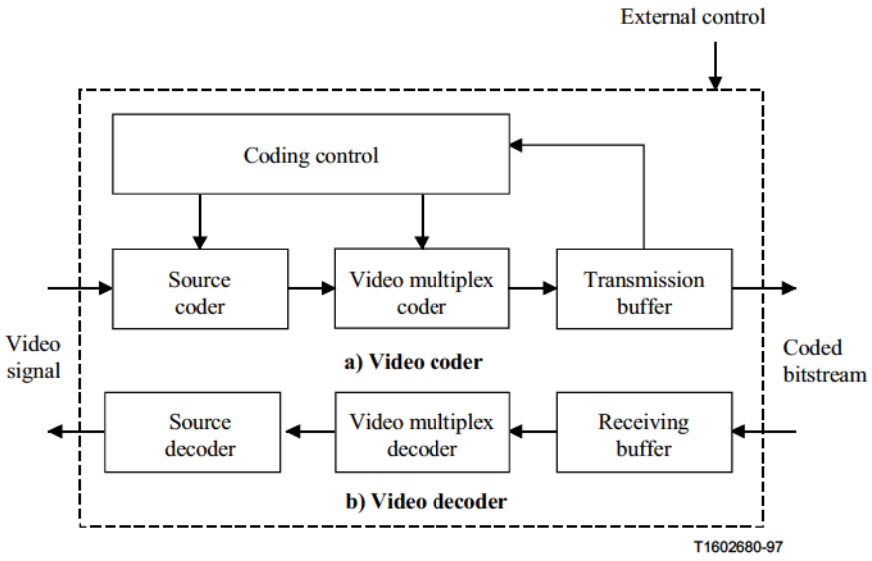



**Figure 1/H.263 – Outline block diagram of the video codec**

### 3.4 Source coding algorithm

A hybrid of inter-picture prediction to utilize temporal redundancy and transform coding of the remaining signal to reduce spatial redundancy is adopted. The decoder has motion compensation capability, allowing optional incorporation of this technique in the coder. Half pixel precision is used for the motion compensation, as opposed to Recommendation H.261 where full pixel precision and a loopfilter are used. Variable length coding is used for the symbols to be transmitted.

In addition to the core H.263 coding algorithm, sixteen negotiable coding options can be used, either together or separately (subject to certain restrictions). Additional supplemental information may also be included in the bitstream for enhanced display capability and for external usage. A forward error correction method for application to the resulting video bitstream is provided for use when necessary. The negotiable coding options, forward error correction, and supplemental information usage are described in the subsequent subclauses.

### 3.4.1 Continuous Presence Multipoint and Video Multiplex mode

In this optional mode, up to four separate video "Sub-Bitstreams" can be sent within the same video channel. This feature is designed for use in continuous presence multipoint application or other situations in which separate logical channels are not available, but the use of multiple video bitstreams is desired (see also Annex C).

### 3.4.2 Unrestricted Motion Vector mode

In this optional mode, motion vectors are allowed to point outside the picture. The edge pixels are used as prediction for the "non-existing" pixels. With this mode a significant gain is achieved if there is movement across the edges of the picture, especially for the smaller picture formats (see also Annex D). Additionally, this mode includes an extension of the motion vector range so that larger motion vectors can be used. This is especially useful in case of camera movement and large picture formats.

### 3.4.3 Syntax-based Arithmetic Coding mode

In this optional mode, arithmetic coding is used instead of variable length coding. The SNR and reconstructed pictures will be the same, but significantly fewer bits will be produced (see also Annex E).

### 3.4.4 Advanced Prediction mode

In this optional mode, Overlapped Block Motion Compensation (OBMC) is used for the luminance part of P-pictures (see also Annex F). Four $8 \times 8$ vectors instead of one $16 \times 16$ vector are used for some of the macroblocks in the picture. The encoder has to decide which type of vectors to use. Four vectors use more bits, but give better prediction. The use of this mode generally gives a considerable improvement. A subjective gain is achieved because OBMC results in less blocking artifacts.

### 3.4.5 PB-frames mode

A PB-frame consists of two pictures being coded as one unit. The name PB comes from the name of picture types in Recommendation H.262 where there are P-pictures and B-pictures. Thus a PB-frame consists of one P-picture which is predicted from the previous decoded P-picture and one B-picture which is predicted from both the previous decoded P-picture and the P-picture currently being decoded. The name B-picture was chosen because parts of B-pictures may be bidirectionally predicted from the past and future pictures. With this coding option, the picture rate can be increased considerably without substantially increasing the bit rate (see also Annex G). However, an Improved PB-frames mode is also provided (see also Annex M). The original PB-frames mode is retained herein only for purposes of compatibility with systems made prior to the adoption of the Improved PB-frames mode.

### 3.4.6 Forward Error Correction

A forward error correction method is specified for use when necessary for the protection of the video bitstream for use when appropriate. The method provided for forward error correction is the same BCH code method specified also in Recommendation H.261 (see also Annex H).

### 3.4.7 Advanced INTRA Coding mode

In this optional mode, INTRA blocks are first predicted from neighbouring INTRA blocks prior to coding (see also Annex I). Separate Variable Length Code (VLC) tables are defined for the INTRA blocks. The technique is applied to INTRA-macroblocks within INTRA pictures and to INTRA-macroblocks within INTER pictures. This mode significantly improves the compression performance over the INTRA coding of the core H.263 syntax.

### 3.4.8 Deblocking Filter mode

In this optional mode, a filter is applied across the 8 × 8 block edge boundaries of decoded I- and P-pictures to reduce blocking artifacts (see also Annex J). The purpose of the filter is to mitigate the occurrence of block edge artifacts in the decoded picture. The filter affects the picture that is used for the prediction of subsequent pictures and thus lies within the motion prediction loop.

### 3.4.9 Slice Structured mode

In this optional mode, a "slice" layer is substituted for the GOB layer of the bitstream syntax (see also Annex K). The purposes of this mode are to provide enhanced error resilience capability, to make the bitstream more amenable to use with an underlying packet transport delivery, and to minimize video delay. A slice is similar to a GOB, in that it is a layer of the syntax that lies between the picture layer and the macroblock layer. However, the use of a slice layer allows a flexible partitioning of the picture, in contrast with the fixed partitioning and fixed transmission order required by the GOB structure.

### 3.4.10 Supplemental enhancement information

Additional supplemental information may be included in the bitstream to signal enhanced display capability or to provide information for external usage (see also Annex L). This supplemental information can be used to signal a full-picture or partial-picture freeze or freeze-release request with or without resizing, can also be used to tag particular pictures or sequences of pictures within the video stream for external usage, and can be used to convey chroma key information for video compositing. The supplemental information may be present in the bitstream even though the decoder may not be capable of providing the enhanced capability to use it, or even to properly interpret it – simply discarding the supplemental information is allowable by decoders unless a requirement to provide the requested capability has been negotiated by external means.

### 3.4.11 Improved PB-frames mode

This optional mode represents an improvement compared to the PB-frames mode option (see also Annexes G and M). The main difference between the two modes is that in the Improved PB-frames mode, each B-block may be forward predicted using a separate motion vector or backward predicted with a zero vector. This significantly improves coding efficiency in situations in which downscaled P-vectors are not good candidates for B-prediction. The backward prediction is particularly useful when there is a scene cut between the previous P-frame and the PB-frame.

### 3.4.12 Reference Picture Selection mode

An optional mode is provided which improves the performance of real-time video communication over error-prone channel by allowing temporal prediction from pictures other than the most recently-sent reference picture (see also Annex N). This mode can be used with back-channel status messages which are sent back to the encoder to inform it about whether its bitstream is being properly received. In error-prone channel environments, this mode allows the encoder to optimize its video encoding for the conditions of the channel.

### 3.4.13 Temporal, SNR and Spatial Scalability mode

In this optional mode, there is support for temporal, SNR, and spatial scalability (see also Annex O). Scalability implies that a bitstream is composed of a base layer and one or more associated enhancement layers. The base layer is a separately decodable bitstream. The enhancement layers can be decoded in conjunction with the base layer to increase perceived quality by either increasing the picture rate, increasing the picture quality, or increasing the picture size. SNR scalability refers to enhancement information to increase the picture quality without increasing picture resolution. Spatial scalability refers to enhancement information to increase the picture quality by increasing picture resolution either horizontally, vertically, or both. There is also support for temporal scalability by the use of B-pictures. A B-picture is a scalability enhancement containing pictures that can be bidirectionally predicted from two pictures in a reference layer, one temporally previous to the current picture and one temporally subsequent. B-pictures allows enhancement layer information to be used to increase perceived quality by increasing the picture rate of the displayed enhanced video sequence. This mode can be useful for heterogenous networks with varying bandwidth capacity and also in conjunction with error correction schemes.

### 3.4.14 Reference Picture Resampling mode

A syntax is provided for supporting an optional mode for which the reference picture used for video image prediction is processed by a resampling operation prior to its use in forming a predictor for the current input picture (see also Annex P). This allows efficient dynamic selection of an appropriate image resolution for video encoding, and can also support picture warping for use as a global motion compensator or special-effect generator.

### 3.4.15 Reduced-Resolution Update mode

An optional mode is provided which allows reduced-resolution updates to a reference picture having a higher resolution (see also Annex Q). This mode is expected to be used when encoding a highly active scene, and allows an encoder to increase the picture rate at which moving parts of a scene can be represented, while maintaining a higher resolution representation in more static areas of the scene.

### 3.4.16 Independent Segment Decoding mode

An optional mode is provided which allows a picture to be constructed without any data dependencies which cross GOB or multi-GOB video picture segments or slice boundaries (see also Annex R). This mode provides error robustness by preventing the propagation of erroneous data across the boundaries of the video picture segment areas.

### 3.4.17 Alternative INTER VLC mode

An optional mode is provided which improves the efficiency of INTER picture coding when significant changes are evident in the picture (see also Annex S). This efficiency improvement is obtained by allowing a VLC code originally designed for INTRA pictures to be used for some INTER picture coefficients as well.

### 3.4.18 Modified Quantization mode

An optional mode is provided which improves the bit-rate control ability for encoding, reduces chrominance quantization error, expands the range of representable DCT coefficients, and places certain restrictions on coefficient values (see also Annex T). This mode modifies the semantics of the differential quantization step size parameter of the bitstream by broadening the range of step size changes that can be specified. It also reduces the quantization step size used for chrominance data. The range of DCT coefficient levels is broadened to ensure that any possible coefficient value can be

encoded to within the accuracy allowed by the step size. Certain restrictions are also placed on coefficients in this mode to increase error detection performance and minimize decoder complexity.

### 3.5 Bit rate

The transmission clock is provided externally. The video bit rate may be variable. In this Recommendation no constraints on the video bit rate are given; constraints will be given by the terminal or the network.

### 3.6 Buffering

The encoder shall control its output bitstream to comply with the requirements of the hypothetical reference decoder defined in Annex B. Video data shall be provided on every valid clock cycle. This can be ensured by MCBPC stuffing (see Tables 7 and 8) or, when forward error correction is used, also by forward error correction stuffing frames (see Annex H).

The number of bits created by coding any single picture shall not exceed a maximum value specified by the parameter BPPmaxKb which is measured in units of 1024 bits. The minimum allowable value of the BPPmaxKb parameter depends on the largest picture size that has been negotiated for use in the bitstream (see Table 1). The picture size is measured as the picture width times height for the luminance (Y) component, measured in pixels. An encoder may use a larger value for BPPmaxKb than as specified in Table 1, provided the larger value is first negotiated by external means, for example Recommendation H.245.

When the Temporal, SNR, and Spatial Scalability mode (Annex O) is employed, the number of bits sent for each picture in each enhancement layer shall not exceed the maximum value specified by BPPmaxKb.

**Table 1/H.263 – Minimum BPPmaxKb for different source picture formats**

| Y picture size in pixels | Minimum BPPmaxKb |
|---|---|
| up to 25 344 (or QCIF) | 64 |
| 25 360 to 101 376 (or CIF) | 256 |
| 101 392 to 405 504 (or 4CIF) | 512 |
| 405 520 and above | 1024 |

### 3.7 Symmetry of transmission

The codec may be used for bidirectional or unidirectional visual communication.

### 3.8 Error handling

Error handling should be provided by external means (for example, Recommendation H.223). If it is not provided by external means (for example, in Recommendation H.221) the optional error correction code and framing as described in Annex H can be used.

A decoder can send a command to encode one or more GOBs (or slices if Annex K is employed) of its next picture in INTRA mode with coding parameters such as to avoid buffer overflow. A decoder can also send a command to transmit only non-empty GOB headers if the Slice Structured mode (see Annex K) is not in use. The transmission method for these signals is by external means (for example, Recommendation H.245).

### 3.9 Multipoint operation

Features necessary to support switched multipoint operation are included in Annex C.

## 4 Source coder

### 4.1 Source format

The source coder operates on non-interlaced pictures having a source format which is defined in terms of:

1) the picture format, as determined by the number of pixels per line, the number of lines per picture, and the pixel aspect ratio; and

2) the timing between pictures, as determined by the Picture Clock Frequency (PCF). For example, the Common Intermediate Format (CIF) has 352 pixels per line, 288 lines, a pixel aspect ratio of 12:11, and a picture clock frequency of 30 000/1001 pictures per second.

The source coder operates on non-interlaced pictures occurring at a Picture Clock Frequency (PCF) of 30 000/1001 (approximately 29.97) times per second, termed the CIF PCF. It is also possible to negotiate the use of an optional custom PCF by external means. A custom PCF is given by 1 800 000/(clock divisor * clock conversion factor) where clock divisor can have values of 1 through 127 and clock conversion factor can either be 1000 or 1001. The tolerance on the picture clock frequency is ± 50 ppm.

Pictures are coded as luminance and two colour difference components (Y, $C_B$ and $C_R$). These components and the codes representing their sampled values are as defined in ITU-R Recommendation BT.601-5.

- Black = 16;
- White = 235;
- Zero colour difference = 128;
- Peak colour difference = 16 and 240.

These values are nominal ones and the coding algorithm functions with input values of 1 through to 254.

There are five standardized picture formats: sub-QCIF, QCIF, CIF, 4CIF and 16CIF. It is also possible to negotiate a custom picture format. For all of these picture formats, the luminance sampling structure is dx pixels per line, dy lines per picture in an orthogonal arrangement. Sampling of each of the two colour difference components is at dx/2 pixels per line, dy/2 lines per picture, orthogonal. The values of dx, dy, dx/2 and dy/2 are given in Table 2 for each of the standardized picture formats.

**Table 2/H.263 – Number of pixels per line and number of lines for each of the standardized H.263 picture formats**

| Picture format | Number of pixels for luminance (dx) | Number of lines for luminance (dy) | Number of pixels for chrominance (dx/2) | Number of lines for chrominance (dy/2) |
|---|---|---|---|---|
| sub-QCIF | 128 | 96 | 64 | 48 |
| QCIF | 176 | 144 | 88 | 72 |
| CIF | 352 | 288 | 176 | 144 |
| 4CIF | 704 | 576 | 352 | 288 |
| 16CIF | 1408 | 1152 | 704 | 576 |

For all picture formats, colour difference samples are sited such that their block boundaries coincide with luminance block boundaries as shown in Figure 2. The pixel aspect ratio is the same for each of the standardised picture formats and is the same as defined for QCIF and CIF in Recommendation H.261: (288/3):(352/4), which simplifies to 12:11 in relatively prime numbers. The picture area covered by all of the standardized picture formats except the sub-QCIF picture format has an aspect ratio of 4:3.

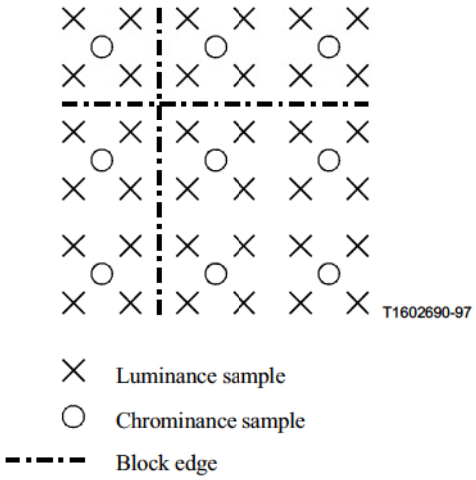



**Figure 2/H.263 – Positioning of luminance and chrominance samples**

Custom picture formats can have a custom pixel aspect ratio as described in Table 3, if the custom pixel aspect ratio use is first negotiated by external means. Custom picture formats can have any number of lines and any number of pixels per line, provided that the number of lines is divisible by four and is in the range [4, ..., 1152], and provided that the number of pixels per line is also divisible by four and is in the range [4, ..., 2048]. For picture formats having a width or height that is not divisible by 16, the picture is decoded in the same manner as if the width or height had the next larger size that would be divisible by 16 and then the picture is cropped at the right and the bottom to the proper width and height for display purposes only.

**Table 3/H.263 – Custom pixel aspect ratios**

| Pixel aspect ratio | Pixel width:Pixel height |
|---|---|
| Square | 1:1 |
| CIF | 12:11 |
| 525-type for 4:3 picture | 10:11 |
| CIF for 16:9 picture | 16:11 |
| 525-type for 16:9 picture | 40:33 |
| Extended PAR | m:n, m and n are relatively prime |

All decoders and encoders shall be able to operate using the CIF picture clock frequency. Some decoders and encoders may also support custom picture clock frequencies. All decoders shall be able to operate using the sub-QCIF picture format. All decoders shall also be able to operate using the QCIF picture format. Some decoders may also operate with CIF, 4CIF or 16CIF, or custom picture formats. Encoders shall be able to operate with one of the picture formats sub-QCIF and QCIF. The

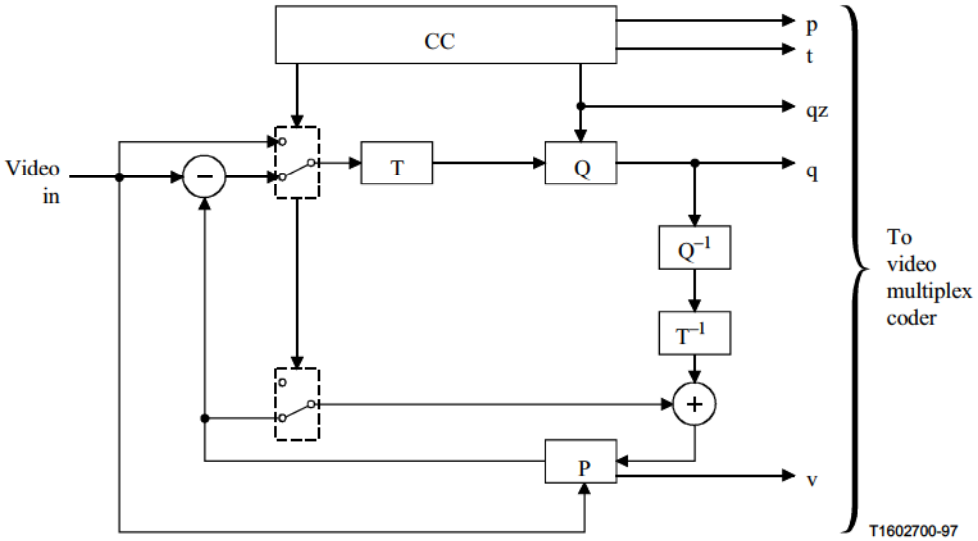
encoders determine which of these two formats are used, and are not obliged to be able to operate with both. Some encoders can also operate with CIF, 4CIF, 16CIF, or custom picture formats. Which optional formats and which picture clock frequencies can be handled by the decoder is signalled by external means, for example Recommendation H.245. For a complete overview of possible picture formats and video coding algorithms, refer to the terminal description, for example Recommendation H.324.

NOTE – For CIF, the number of pixels per line is compatible for practical purposes with sampling the active portions of the luminance and colour difference signals from 525- or 625-line sources at 6.75 and 3.375 MHz respectively. These frequencies have a simple relationship to those in ITU-R Recommendation BT.601-5.

Means shall be provided to restrict the maximum picture rate of encoders by having a minimum number of non-transmitted pictures between transmitted ones. Selection of this minimum number shall be by external means (for example, Recommendation H.245). For the calculation of the minimum number of non-transmitted pictures in PB-frames mode, the P-picture and the B-picture of a PB-frames unit are taken as two separate pictures.

## 4.2 Video source coding algorithm

The source coder is shown in generalized form in Figure 3. The main elements are prediction, block transformation and quantization.




T Transform
Q Quantizer
P Picture Memory with motion compensated variable delay
CC Coding control
p Flag for INTRA/INTER
t Flag for transmitted or not
qz Quantizer indication
q Quantizing index for transform coefficients
v Motion vector

**Figure 3/H.263 – Source coder**

### 4.2.1 GOBs, slices, macroblocks and blocks

Each picture is divided either into Groups Of Blocks (GOBs) or into slices.

A Group Of Blocks (GOB) comprises of up to k * 16 lines, where k depends on the number of lines in the picture format and depends on whether the optional Reduced-Resolution Update mode is in use (see Annex Q). The dependencies are shown in Table 4. If the number of lines is less than or equal to 400 and the optional Reduced-Resolution Update mode is not in use, then k = 1. If the number of lines is less than or equal to 800 and the optional Reduced-Resolution Update mode is in use or the number of lines is more than 400, then k = 2. If the number of lines is more than 800, then k = 4. When using custom picture sizes, the number of lines in the last (bottom-most) GOB may be less than k * 16 if the number of lines in the picture is not divisible by k * 16. However, every GOB in each of the standardized picture formats has k * 16 lines, as the number of lines in each standardized picture format is an integer multiple of k * 16. Thus, for example, if the optional Reduced-Resolution mode is not in use, the number of GOBs per picture is 6 for sub-QCIF, 9 for QCIF, and 18 for CIF, 4CIF and 16CIF. The GOB numbering is done by use of vertical scan of the GOBs, starting with the upper GOB (number 0) and ending with the bottom-most GOB. An example of the arrangement of GOBs in a picture is given for the CIF picture format in Figure 4. Data for each GOB consists of a GOB header (may be empty) followed by data for macroblocks. Data for GOBs is transmitted per GOB in increasing GOB number.

**Table 4/H.263 – Parameter k for GOB size definition**

| Number of Lines dy | Value of k when not in RRU mode | Value of k when in RRU mode |
|---|---|---|
| 4,…,400 | 1 | 2 |
| 404,…,800 | 2 | 2 |
| 804,…,1152 | 4 | 4 |

The Slice Structured mode is described in Annex K. Slices are similar to GOBs in that they are a multi-macroblock layer of the syntax, but slices have a more flexible shape and usage than GOBs, and slices may appear in the bitstream in any order, under certain conditions.

| |
|---|
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 11 |
| 12 |
| 13 |
| 14 |
| 15 |
| 16 |
| 17 |

**Figure 4/H.263 – Arrangement of Groups of Blocks in a CIF picture**

Each GOB is divided into macroblocks. The macroblock structure depends on whether the optional Reduced-Resolution Update (RRU) mode is in use (see Annex Q). Unless in RRU mode, each macroblock relates to 16 pixels by 16 lines of Y and the spatially corresponding 8 pixels by 8 lines of $C_B$ and $C_R$. Further, a macroblock consists of four luminance blocks and the two spatially corresponding colour difference blocks as shown in Figure 5. Each luminance or chrominance block thus relates to 8 pixels by 8 lines of Y, $C_B$ or $C_R$. Unless in RRU mode, a GOB comprises one macroblock row for sub-QCIF, QCIF and CIF, two macroblock rows for 4CIF and four macroblock rows for 16CIF.

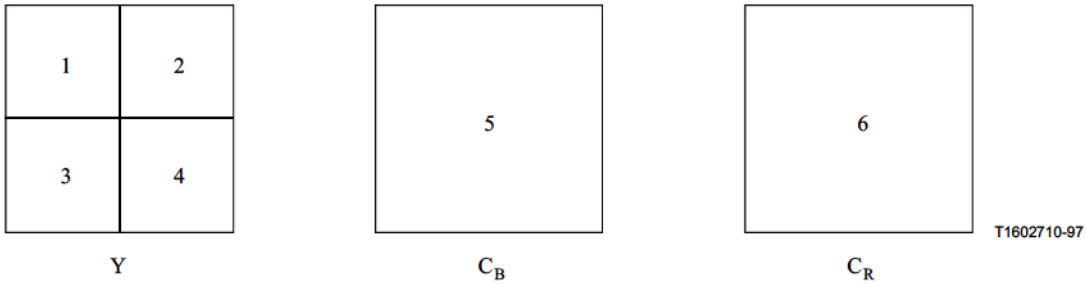



**Figure 5/H.263 – Arrangement of blocks in a macroblock**

When in RRU mode, a macroblock relates to 32 pixels by 32 lines of Y and the spatially corresponding 16 pixels by 16 lines of $C_B$ and $C_R$, and each luminance or chrominance block relates to 16 pixels by 16 lines of Y, $C_B$ or $C_R$. Furthermore, a GOB comprises one macroblock row for CIF and 4CIF, and two macroblock rows for 16CIF.

The macroblock numbering is done by use of horizontal scan of the macroblock rows from left to right, starting with the upper macroblock row and ending with the lower macroblock row. Data for the macroblocks is transmitted per macroblock in increasing macroblock number. Data for the blocks is transmitted per block in increasing block number (see Figure 5).

The criteria for choice of mode and transmitting a block are not subject to recommendation and may be varied dynamically as part of the coding control strategy. Transmitted blocks are transformed and resulting coefficients are quantized and entropy coded.

### 4.2.2 Prediction

The primary form of prediction is inter-picture and may be augmented by motion compensation (see 4.2.3). The coding mode in which temporal prediction is applied is called INTER; the coding mode is called INTRA if no temporal prediction is applied. The INTRA coding mode can be signalled at the picture level (INTRA for I-pictures or INTER for P-pictures) or at the macroblock level in P-pictures. In the optional PB-frames mode, B-pictures are always coded in INTER mode. The B-pictures are partly predicted bidirectionally (refer to Annex G).

In total, H.263 has seven basic picture types (of which only the first two are mandatory) which are defined primarily in terms of their prediction structure:

1) INTRA: A picture having no reference picture(s) for prediction (also called an I-picture);
2) INTER: A picture using a temporally previous reference picture (also called a P-picture);
3) PB: A frame representing two pictures and having a temporally previous reference picture (see Annex G);
4) Improved PB: A frame functionally similar but normally better than a PB-frame (see Annex M);

5) B: A picture having two reference pictures, one of which temporally precedes the B-picture and one of which temporally succeeds the B-picture and has the same picture size (see Annex O);

6) EI: A picture having a temporally simultaneous reference picture which has either the same or a smaller picture size (see Annex O); and

7) EP: A picture having two reference pictures, one of which temporally precedes the EP-picture and one of which is temporally simultaneous and has either the same or a smaller picture size (see Annex O).

As used herein, a "reference" or "anchor" picture is a picture that contains data which can be used by reference as a basis for the decoding of another picture. This use by reference is also known as "prediction", although it sometimes may actually indicate use in a reverse-temporal direction.

### 4.2.3 Motion compensation

The decoder will accept one vector per macroblock or, if the Advanced Prediction mode or the Deblocking Filter mode is used, one or four vectors per macroblock (see Annexes F and J). If the PB-frames mode is used, one additional delta vector can be transmitted per macroblock for adaptation of the motion vectors for prediction of the B-macroblock. Similarly, an Improved PB-frame macroblock (see Annex M) can include an additional forward motion vector. B-picture macroblocks (see Annex O) can be transmitted together with a forward and a backward motion vector, and EP-pictures can be transmitted with a forward motion vector.

Both horizontal and vertical components of the motion vectors have integer or half integer values. In the default prediction mode, these values are restricted to the range [−16, 15.5] (this is also valid for the forward and backward motion vector components for B-pictures).

In the Unrestricted Motion Vector mode, however, the maximum range for vector components is increased. If PLUSPTYPE is absent, the range is [−31.5, 31.5], with the restriction that only values that are within a range of [−16, 15.5] around the predictor for each motion vector component can be reached if the predictor is in the range [−15.5, 16]. If PLUSPTYPE is absent and the predictor is outside [−15.5, 16], all values within the range [−31.5, 31.5] with the same sign as the predictor plus the zero value can be reached. If PLUSPTYPE is present, the motion vector values are less restricted (see also Annex D).

In the Reduced-Resolution Update mode, the motion vector range is enlarged to approximately double size, and each vector component is restricted to have only a half-integer or zero value. Therefore, the range of each motion vector component is [−31.5, 30.5] in the default Reduced-Resolution Update mode (see Annex Q) and a larger range if the Unrestricted Motion Vector mode is also used (see also Annex D).

A positive value of the horizontal or vertical component of the motion vector signifies that the prediction is formed from pixels in the referenced picture which are spatially to the right or below the pixels being predicted.

Motion vectors are restricted such that all pixels referenced by them are within the coded picture area, except when the Unrestricted Motion Vector mode, the Advanced Prediction mode, the Advanced Prediction mode, or the Deblocking Filter mode is used (see Annexes D, F, and J), or in B- and EP-pictures of the Temporal, SNR, and Spatial Scalability mode (see Annex O).

### 4.2.4 Quantization

Unless the optional Advanced INTRA Coding mode or Modified Quantization mode is in use, the number of quantizers is 1 for the first coefficient of INTRA blocks and 31 for all other coefficients. Within a macroblock the same quantizer is used for all coefficients except the first one of INTRA

blocks. The decision levels are not defined. The first coefficient of INTRA blocks is nominally the transform dc value uniformly quantized with a step size of 8. Each of the other 31 quantizers use equally spaced reconstruction levels with a central dead-zone around zero and with a step size of an even value in the range 2 to 62. For the exact formulas, refer to 6.2. For quantization using the Advanced INTRA Coding mode, refer to Annex I. For quantization using the Modified Quantization mode, refer to Annex T.

NOTE – For the smaller quantization step sizes, the full dynamic range of the transform coefficients cannot be represented unless the optional Modified Quantization mode is in use.

### 4.3 Coding control

Several parameters may be varied to control the rate of generation of coded video data. These include processing prior to the source coder, the quantizer, block significance criterion and temporal subsampling. The proportions of such measures in the overall control strategy are not subject to recommendation.

When invoked, temporal subsampling is performed by discarding complete pictures.

A decoder can signal its preference for a certain tradeoff between spatial and temporal resolution of the video signal. The encoder shall signal its default tradeoff at the beginning of the call and shall indicate whether it is capable of responding to decoder requests to change this tradeoff. The transmission method for these signals is by external means (for example, Recommendation H.245).

### 4.4 Forced updating

This function is achieved by forcing the use of the INTRA mode of the coding algorithm. The update pattern is not defined. To control the accumulation of inverse transform mismatch error, each macroblock shall be coded in INTRA mode at least once every 132 times when coefficients are transmitted for this macroblock in P-pictures. A similar requirement applies when using optional EP-pictures (see Annex O), for which each macroblock shall be coded in an INTRA or upward mode at least once every 132 times when coefficients are transmitted for the macroblock.

### 4.5 Byte alignment of start codes

Byte alignment of start codes is achieved by inserting a stuffing codeword consisting of less than 8 zero-bits before the start code such that the first bit of the start code is the first (most significant) bit of a byte. A start code is therefore byte aligned if the position of its most significant bit is a multiple of 8 bits from the first bit in the H.263 bitstream. All picture, slice, and EOSBS start codes shall be byte aligned, and GOB and EOS start codes may be byte aligned.

NOTE 1 – The number of bits spent for a certain picture is variable but always a multiple of 8 bits.

NOTE 2 – H.324 requires H.263 encoders to align picture start codes with the start of logical information units passed to the Adaptation Layer (AL_SDU's).

## 5 Syntax and semantics

The video syntax is arranged in a hierarchical structure with four primary layers. From top to bottom the layers are:

- Picture;
- Group of Blocks, or slice, or video picture segment;
- Macroblock;
- Block.

The syntax diagram is shown in Figure 6. A guide for interpretation of the diagram consists of the following:

1) Arrow paths show the possible flows of syntax elements. Any syntax element which has zero length is considered absent for arrow path diagramming purposes (thus, for example, there is an arrow path bypassing PSTUF despite the mandatory nature the PSTUF field since the length of the PSTUF field may be zero).

2) Abbreviations and semantics for each syntax element are as defined in later clauses.

3) The set of syntax elements and arrow paths shown using thick solid lines denotes the syntax flow of the "baseline" mode of operation, without the use of any optional enhancements. (This syntax was also present in the Version 1 of this Recommendation, and remains unchanged in any way.)

4) The set of syntax elements and arrow paths shown using thick dotted lines denotes the additional elements in the syntax flow of the optional enhancements which have been present in both Version 1 and Version 2 of this Recommendation. (This syntax remains unchanged in any way.)

5) The set of syntax elements and arrow paths shown using thin solid lines denotes the additional new elements in the syntax flow of the optional enhancements which are specific to the additional optional features added in Version 2. (This syntax was not present in Version 1.)

6) Syntax element fields shown with square-edged boundaries indicate fixed-length fields, and those with rounded boundaries indicate variable-length fields. One syntax element (DQUANT) is shown with both types of boundaries, because it can have either a variable or fixed length.

7) A fixed-length field is defined to be a field for which the length of the field is not dependent on the data in the content of the field itself. The length of this field is either always the same, or is determined by the prior data in the syntax flow.

8) The term "layer" is used to refer to any part of the syntax which can be understood and diagrammed as a distinct entity.

Unless specified otherwise, the most significant bit is transmitted first. This is bit 1 and is the left most bit in the code tables in this Recommendation. Unless specified otherwise, all unused or spare bits are set to "1". Spare bits shall not be used until their functions are specified by the ITU-T.

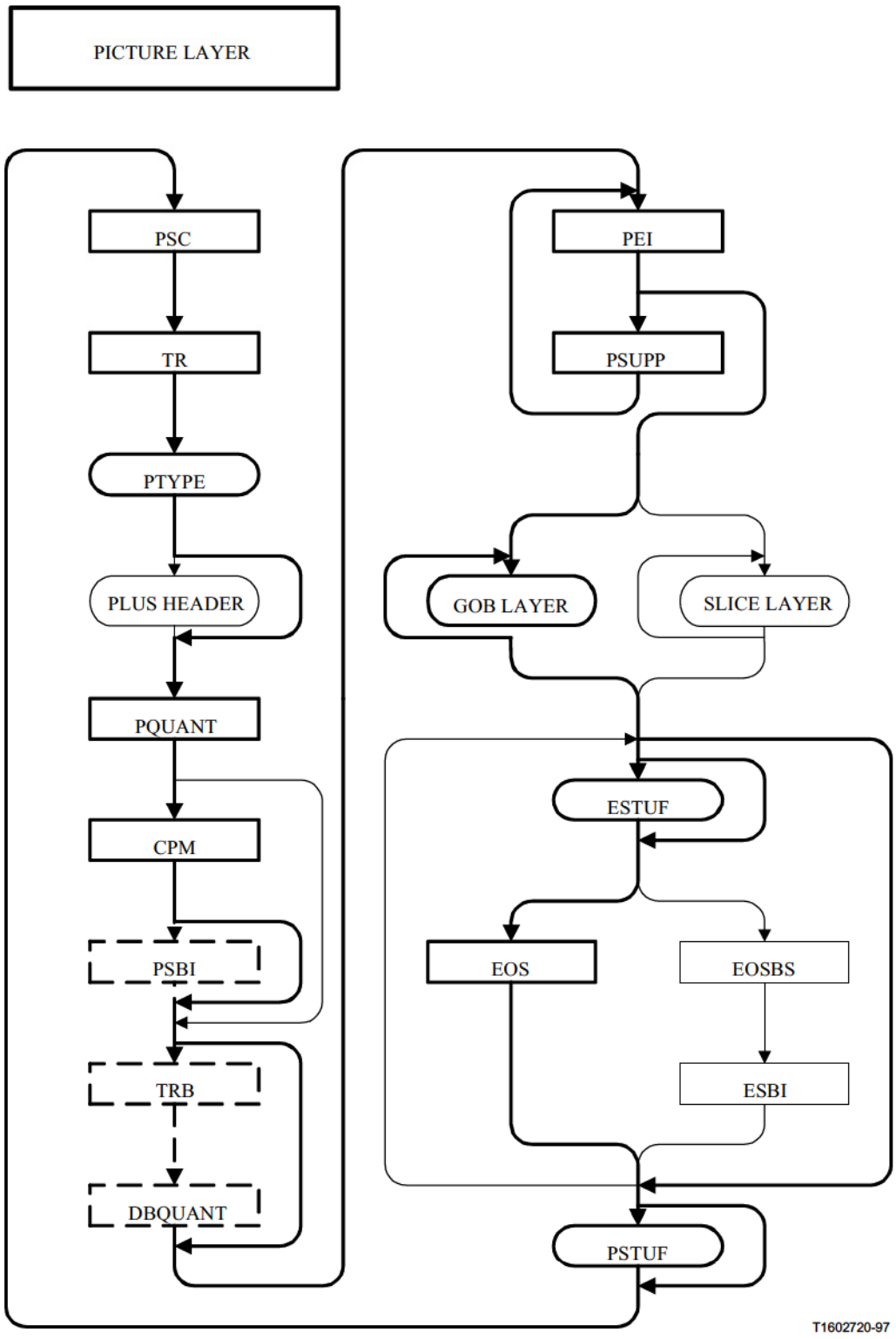



**Figure 6/H.263 (part 1 of 7) – Syntax diagram for the video bitstream**

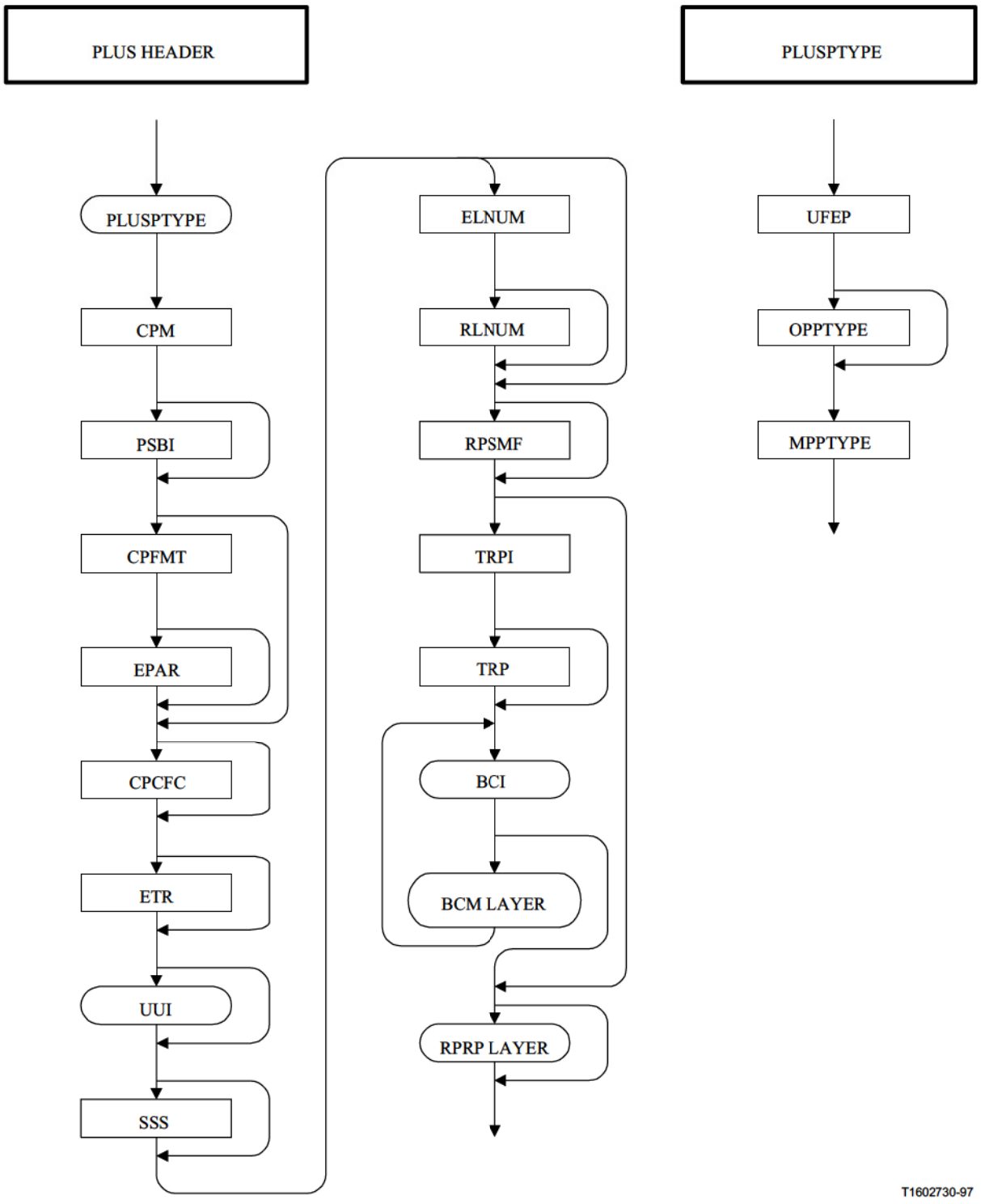



**Figure 6/H.263 (part 2 of 7) – Syntax diagram for the video bitstream**

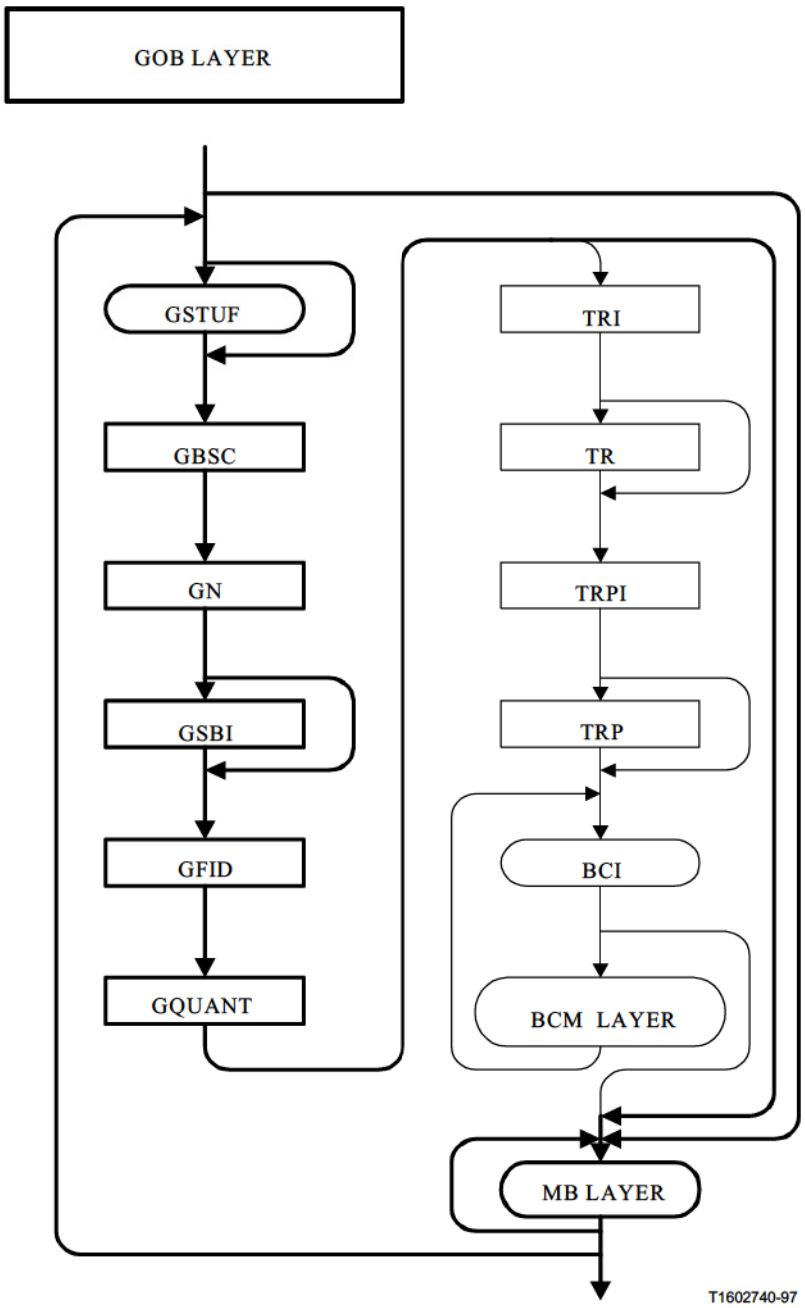



**Figure 6/H.263 (part 3 of 7) – Syntax diagram for the video bitstream**

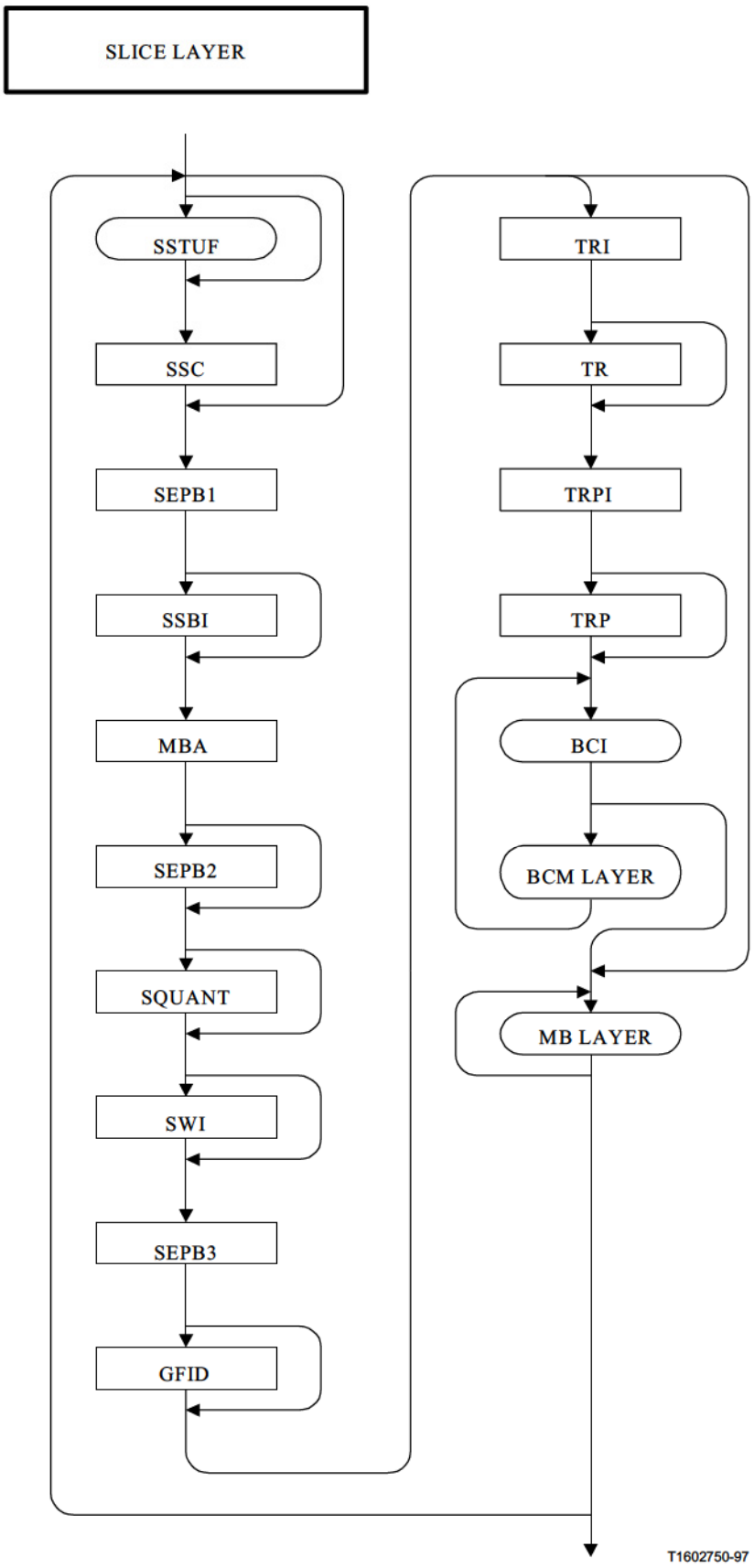



**Figure 6/H.263 (part 4 of 7) – Syntax diagram for the video bitstream**

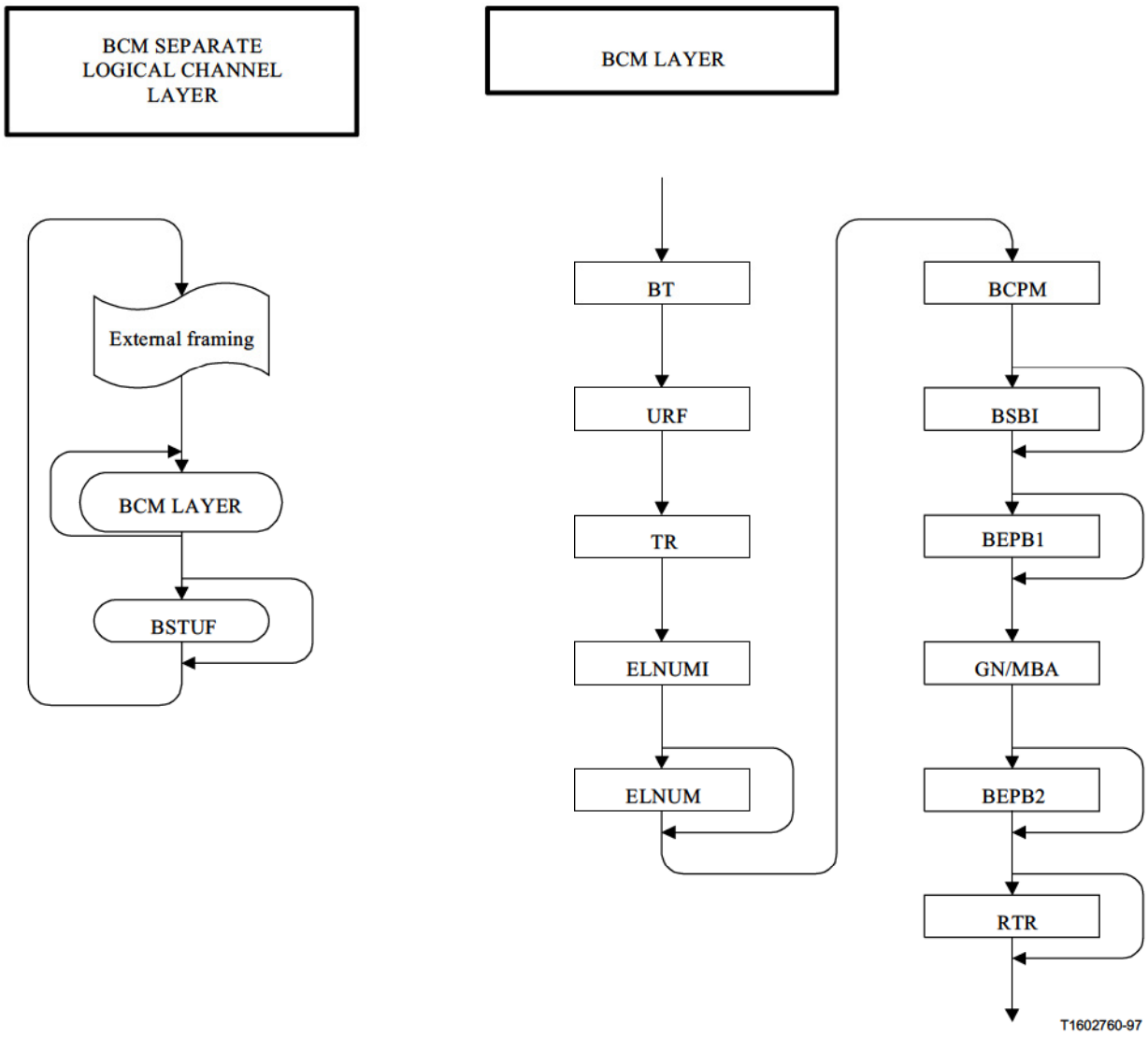



**Figure 6/H.263 (part 5 of 7) – Syntax diagram for the video bitstream**

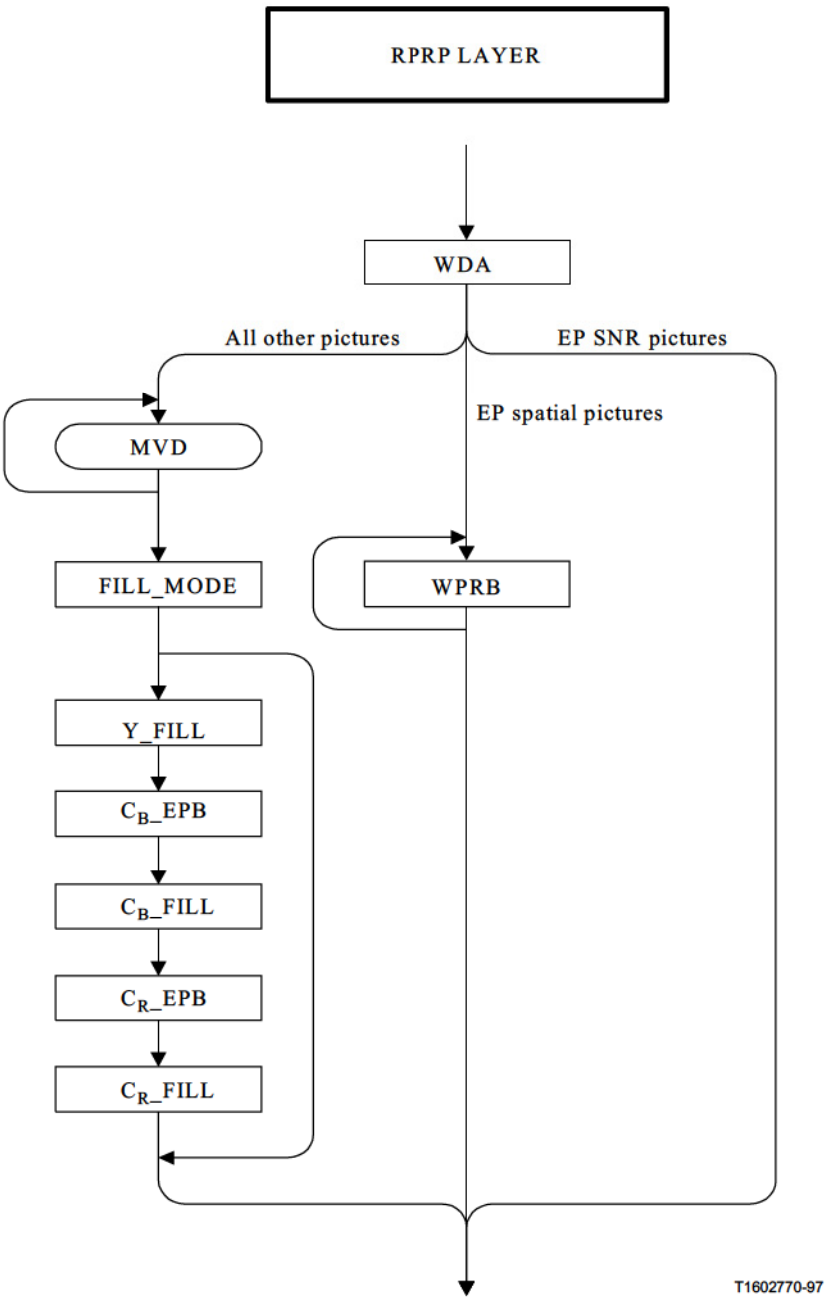



**Figure 6/H.263 (part 6 of 7) – Syntax diagram for the video bitstream**

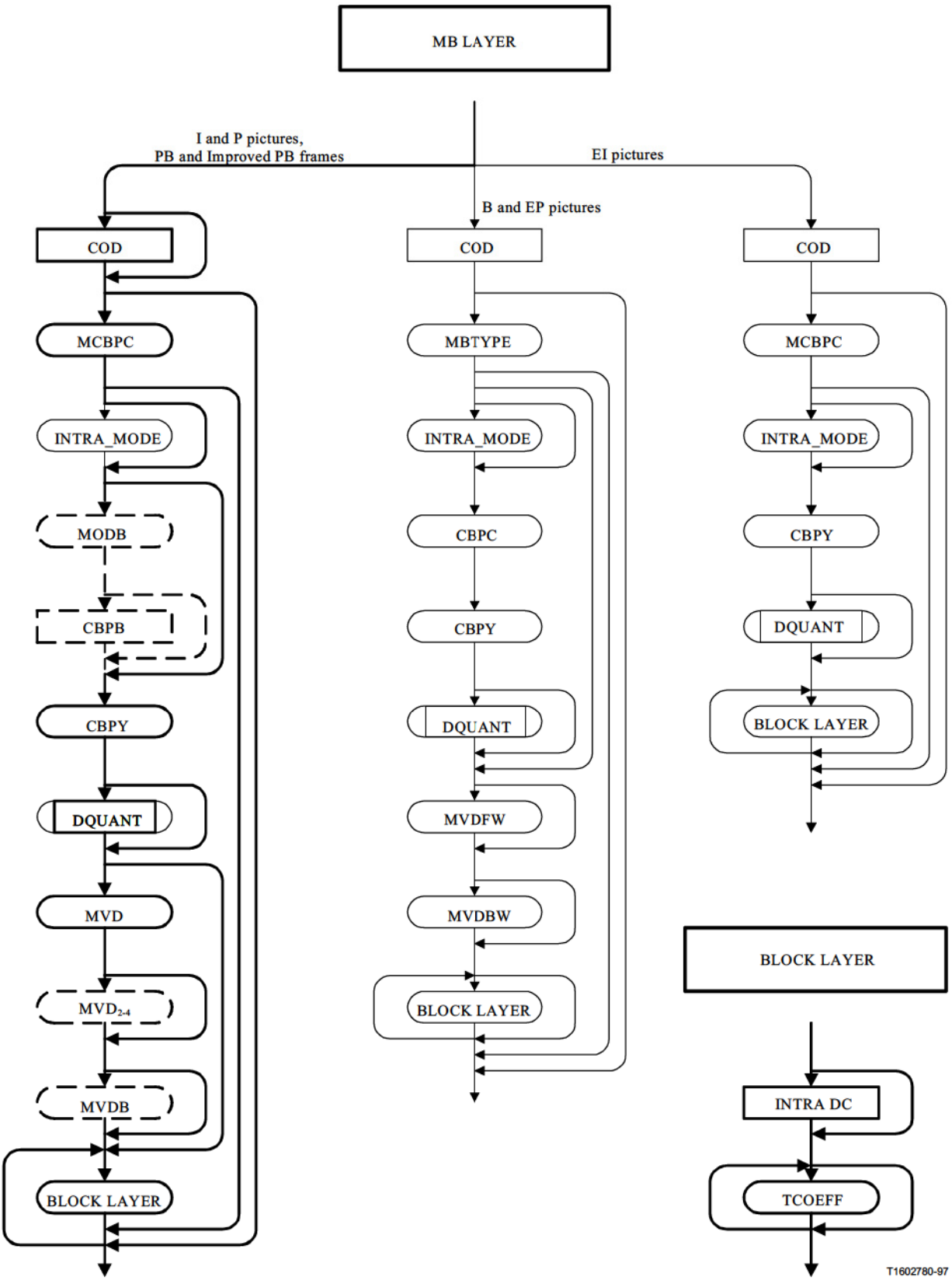



**Figure 6/H.263 (part 7 of 7) – Syntax diagram for the video bitstream**

## 5.1 Picture layer

Data for each picture consists of a picture header followed by data for Group of Blocks or for slices, eventually followed by an optional end-of-sequence code and stuffing bits. The structure is shown in Figure 7 for pictures that do not include the optional PLUSPTYPE data field. PSBI is only present if indicated by CPM. $TR_B$ and DBQUANT are only present if PTYPE indicates use of the "PB-frame" mode (unless the PLUSPTYPE field is present and the use of DBQUANT is indicated therein).

The optional PLUSPTYPE data field is present when so indicated in bits 6-8 of PTYPE. When present, an additional set of data is included in the bitstream, which immediately follows PTYPE and precedes PQUANT. In addition, the CPM and PSBI fields are moved forward in the picture header when PLUSPTYPE is present, so that they appear immediately after PLUSPTYPE rather than being located after PQUANT. The format for the additional data following PLUSPTYPE is shown in Figure 8. All fields in this additional picture header data after PLUSPTYPE are optional, depending on whether their presence is indicated in PLUSPTYPE. When the Slice Structured mode (see Annex K) is in use, slices are substituted for GOBs in the location shown in Figure 7.

Combinations of PSUPP and PEI may not be present, and may be repeated when present. EOS and EOSBS+ESBI may not be present, while ESTUF may be present only if EOS or EOSBS is present. EOS shall not be repeated unless at least one picture start code appears between each pair of EOS codes. Picture headers for dropped pictures are not transmitted.

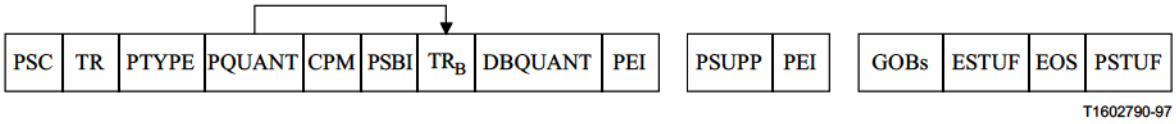



**Figure 7/H.263 – Structure of picture layer (without optional PLUSPTYPE-related fields)**

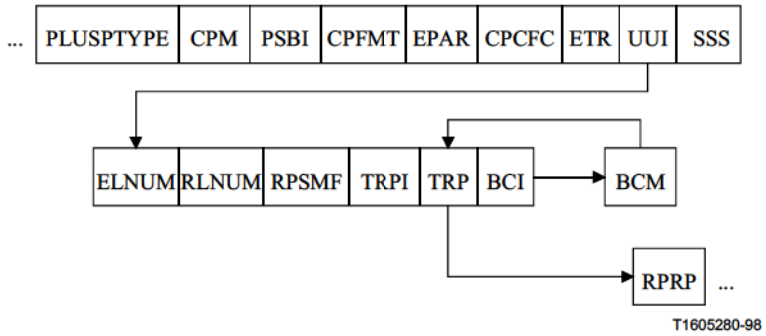



**Figure 8/H.263 – Structure of optional PLUSPTYPE-related fields (located immediately after PTYPE when present)**

### 5.1.1 Picture Start Code (PSC) (22 bits)

PSC is a word of 22 bits. Its value is 0000 0000 0000 0000 1 00000. All picture start codes shall be byte aligned. This shall be achieved by inserting PSTUF bits as necessary before the start code such that the first bit of the start code is the first (most significant) bit of a byte.

### 5.1.2 Temporal Reference (TR) (8 bits)

The value of TR is formed by incrementing its value in the temporally previous reference picture header by one plus the number of skipped or non-reference pictures at the picture clock frequency since the previously transmitted one. The interpretation of TR depends on the active picture clock frequency. Under the standard CIF picture clock frequency, TR is an 8-bit number which can have 256 possible values. The arithmetic is performed with only the eight LSBs. If a custom picture clock frequency is signalled in use, Extended TR in 5.1.8 and TR form a 10-bit number where TR stores the eight Least Significant Bits (LSBs) and ETR stores the two Most Significant Bits (MSBs). The arithmetic in this case is performed with the ten LSBs. In the optional PB-frames or Improved PB-frames mode, TR only addresses P-pictures; for the temporal reference for the B-picture part of PB or Improved PB frames, refer to 5.1.22.

### 5.1.3 Type Information (PTYPE) (Variable Length)

Information about the complete picture:

– Bit 1: Always "1", in order to avoid start code emulation.

– Bit 2: Always "0", for distinction with Recommendation H.261.

– Bit 3: Split screen indicator, "0" off, "1" on.

– Bit 4: Document camera indicator, "0" off, "1" on.

– Bit 5: Full Picture Freeze Release, "0" off, "1" on.

– Bits 6-8: Source Format, "000" forbidden, "001" sub-QCIF, "010" QCIF, "011" CIF, "100" 4CIF, "101" 16CIF, "110" reserved, "111" extended PTYPE.

If bits 6-8 are not equal to "111", which indicates an extended PTYPE (PLUSPTYPE), the following five bits are also present in PTYPE:

– Bit 9: Picture Coding Type, "0" INTRA (I-picture), "1" INTER (P-picture).

– Bit 10: Optional Unrestricted Motion Vector mode (see Annex D), "0" off, "1" on.

– Bit 11: Optional Syntax-based Arithmetic Coding mode (see Annex E), "0" off, "1" on.

– Bit 12: Optional Advanced Prediction mode (see Annex F), "0" off, "1" on.

– Bit 13: Optional PB-frames mode (see Annex G), "0" normal I- or P-picture, "1" PB-frame.

Split screen indicator is a signal that indicates that the upper and lower half of the decoded picture could be displayed side by side. This bit has no direct effect on the encoding or decoding of the picture.

Full Picture Freeze Release is a signal from an encoder which responds to a request for packet retransmission (if not acknowledged) or fast update request (see also Annex C) or picture freeze request (see also Annex L) and allows a decoder to exit from its freeze picture mode and display the decoded picture in the normal manner.

If bits 6-8 indicate a different source format than in the previous picture header, the current picture shall be an I-picture, unless an extended PTYPE is indicated in bits 6-8 and the capability to use the optional Reference Picture Resampling mode (see Annex P) has been negotiated externally (for example Recommendation H.245).

Bits 10-13 refer to optional modes that are only used after negotiation between encoder and decoder (see also Annexes D, E, F and G, respectively). If bit 9 is set to "0", bit 13 shall be set to "0" as well.

Bits 6-8 shall not have a value of "111" which indicates the presence of an extended PTYPE (PLUSPTYPE) unless the capability has been negotiated externally (for example Recommendation H.245) to allow the use of a custom source format or one or more of the other optional modes

available only by the use of an extended PTYPE (see Annexes I through K and M through T). Whenever bit 6-8 do not have a value of "111", all of the additional modes available only by the use of an extended PTYPE shall be considered to have been set to an "off" state and shall be inferred to remain "off" unless explicitly switched on later in the bitstream.

### 5.1.4 Plus PTYPE (PLUSPTYPE) (Variable Length)

A codeword of 12 or 30 bits which is present only if the presence of extended PTYPE is indicated in bits 6-8 of PTYPE.

PLUSPTYPE is comprised of up to three subfields: UFEP, OPPTYPE, and MPPTYPE. OPPTYPE is present only if UFEP has a particular value.

#### 5.1.4.1 Update Full Extended PTYPE (UFEP) (3 bits)

A fixed length codeword of 3 bits which is present only if "extended PTYPE" is indicated in PTYPE bits 6-8. When set to "000", it indicates that only those extended PTYPE fields which need to be signalled in every picture header (MPPTYPE) are included in the current picture header. When set to "001", it indicates that all extended PTYPE fields are included in the current picture header. If the picture type is INTRA or EI, this field shall be set to "001".

In addition, if PLUSPTYPE is present in each of a continuing sequence of pictures, this field shall be set to "001" at least as often as specified by a five-second or five-picture timeout period, whichever allows a larger interval of time. More specifically, the timeout period requires UFEP = "001" to appear in the PLUSPTYPE field (if PLUSPTYPE is present in every intervening picture) of the first picture header with temporal reference indicating a time interval greater than or equal to five seconds since the last occurrence of UFEP = "001", or of the fifth picture after the last occurrence of UFEP = "001" (whichever requirement allows a longer period of time as measured by temporal reference).

Encoders should set UFEP to "001" more often in error-prone environments. Values of UFEP other than "000" and "001" are reserved.

#### 5.1.4.2 The Optional Part of PLUSPTYPE (OPPTYPE) (18 bits)

If UFEP is "001", then the following bits are present in PLUSPTYPE:

- Bits 1-3 Source Format, "000" reserved, "001" sub-QCIF, "010" QCIF, "011" CIF, "100" 4CIF, "101" 16CIF, "110" custom source format, "111" reserved;
- Bit 4 Optional Custom PCF, "0" CIF PCF, "1" custom PCF;
- Bit 5 Optional Unrestricted Motion Vector (UMV) mode (see Annex D), "0" off, "1" on;
- Bit 6 Optional Syntax-based Arithmetic Coding (SAC) mode (see Annex E), "0" off, "1" on;
- Bit 7 Optional Advanced Prediction (AP) mode (see Annex F), "0" off, "1" on;
- Bit 8 Optional Advanced INTRA Coding (AIC) mode (see Annex I), "0" off, "1" on;
- Bit 9 Optional Deblocking Filter (DF) mode (see Annex J), "0" off, "1" on;
- Bit 10 Optional Slice Structured (SS) mode (see Annex K), "0" off, "1" on;
- Bit 11 Optional Reference Picture Selection (RPS) mode (see Annex N), "0" off, "1" on;
- Bit 12 Optional Independent Segment Decoding (ISD) mode (see Annex R), "0" off, "1" on;
- Bit 13 Optional Alternative INTER VLC (AIV) mode (see Annex S), "0" off, "1" on;

– Bit 14 Optional Modified Quantization (MQ) mode (see Annex T), "0" off, "1" on;

– Bit 15 Equal to "1" to prevent start code emulation;

– Bit 16 Reserved, shall be equal to "0";

– Bit 17 Reserved, shall be equal to "0";

– Bit 18 Reserved, shall be equal to "0".

#### 5.1.4.3 The mandatory part of PLUSPTYPE when PLUSPTYPE present (MPPTYPE) (9 bits)

Regardless of the value of UFEP, the following 9 bits are also present in PLUSPTYPE:

– Bits 1-3 Picture Type Code:
"000" I-picture (INTRA),
"001" P-picture (INTER),
"010" Improved PB-frame (see Annex M),
"011" B-picture (see Annex O),
"100" EI-picture (see Annex O),
"101" EP-picture (see Annex O),
"110" Reserved,
"111" Reserved;

– Bit 4 Optional Reference Picture Resampling (RPR) mode (see Annex P), "0" off, "1" on;

– Bit 5 Optional Reduced-Resolution Update (RRU) mode (see Annex Q), "0" off, "1" on;

– Bit 6 Rounding Type (RTYPE) (see 6.1.2);

– Bit 7 Reserved, shall be equal to "0";

– Bit 8 Reserved, shall be equal to "0";

Bit 9 Equal to "1" to prevent start code emulation.

The encoder should control the rounding type so that P-pictures, Improved PB-frames, and EP-pictures have different values for bit 6 (Rounding Type for P-pictures) from their reference pictures for motion compensation. Bit 6 can have an arbitrary value if the reference picture is an I- or an EI-picture. Bit 6 can be set to "1" only when bits 1-3 indicate a P-picture, Improved PB-frame, or EP-picture. For other types of pictures, this bit shall always be set to "0".

#### 5.1.4.4 The semantics of PLUSPTYPE

The mandatory part of PLUSPTYPE consists of features which are likely to be changed on a picture-by-picture basis. Primarily, these are the bits that indicate the picture type among I, P, Improved PB, B, EI, and EP. (Note that the PB-frames mode of Annex G cannot be used with PLUSPTYPE present – the Improved PB-frames mode of Annex M should be used instead.) However, they also include indications for the use of the RPR and RRU modes, as these may change from picture to picture as well.

Features that are likely to remain in use rather than being changed from one picture to another (except in obvious ways as discussed in 5.1.4.5) have been placed in the optional part of PLUSPTYPE. When UFEP is 000, the missing mode information is inferred from the type of picture and from the mode information sent in a previous PLUSPTYPE with UFEP equal to 001.

If PLUSPTYPE is present but UFEP is 000, then:

1) For a P-picture or Improved PB-frame (see Annex M), the pixel aspect ratio, picture width, and picture height are unchanged from those of the reference picture.

2) For a temporal-scalability B-picture (see Annex O) in an enhancement layer, the Reference Layer Number (RLNUM) is the same as the Enhancement Layer Number (ELNUM) if the last picture sent in the enhancement layer was an EI- or EP-picture. If the last picture sent in the enhancement layer was a B-picture, the reference layer number is the same as the reference layer number of the last B-picture. The pixel aspect ratio, picture width and picture height are unchanged from those of the temporally subsequent reference layer picture.

   Note that if temporally surrounding EI- or EP-pictures exist in the same enhancement layer as the B-picture, RLNUM (explicit or implied) shall always be equal to ELNUM. Note also that the pixel aspect ratio, picture width and picture height of a B-picture (explicit or implied) shall always be equal to those of its temporally subsequent reference layer picture.

3) For a SNR/Spatial scalability EP-picture (see Annex O), the pixel aspect ratio, picture width and picture height are unchanged from those of the temporally previous reference picture in the same enhancement layer.

#### 5.1.4.5 Mode restrictions for certain picture types and mode inference rules

Certain modes do not apply to certain types of pictures. Particularly, these restrictions apply:

1) The following modes do not apply within I (INTRA) pictures: Unrestricted Motion Vector (see Annex D), Advanced Prediction (see Annex F), Alternative INTER VLC (see Annex S), Reference Picture Resampling (see Annex P), and Reduced-Resolution Update (see Annex Q).

2) The following modes do not apply within B-pictures (see Annex O): Syntax-based Arithmetic Coding (see Annex E), Deblocking Filter (see Annex J), and Advanced Prediction (see Annex F).

3) The following modes do not apply within EI-pictures (see Annex O): Unrestricted Motion Vector (see Annex D), Syntax-based Arithmetic Coding (see Annex E), Advanced Prediction (see Annex F), Reference Picture Resampling (see Annex P), Reduced-Resolution Update (see Annex Q), and Alternative INTER VLC (see Annex S).

4) The following modes do not apply within EP-pictures (see Annex O): Syntax-based Arithmetic Coding (see Annex E) and Advanced Prediction (see Annex F).

One or more of the modes listed in the above four-item list may have a mode flag having a value "1" in the optional part of PLUSPTYPE within a picture of a type that is prohibited for that mode (types I, B, EI, or EP). This condition is allowed and shall be interpreted subject to the mode inference rules which follow in the next paragraph.

Mode states are subject to the following mode inference rules:

1) Once a mode flag has been set to "1" in the optional part of PLUSPTYPE, the current picture and each subsequent picture in the bitstream shall be assigned a state of "on" for that mode.

2) An inferred state of "off" shall be assigned to any mode which does not apply within a picture having the current picture type code. However, each subsequent picture in the bitstream shall have an inferred state of "on" for that mode (unless this too results in an obvious conflict – which shall be resolved in the same way). In the case of layered scalable bitstreams (see Annex O), the mode state shall be inferred only from within the same layer of the bitstream.

3) The inference of state shall continue until a picture in the same layer that either contains the optional part of PLUSPTYPE or does not contain PLUSPTYPE at all is sent. If a new picture containing the optional part of PLUSPTYPE is sent, the state sent in the new message shall override the old state. If a picture is sent which does not contain PLUSPTYPE (a picture in which bits 6-8 of PTYPE is not "111"), a state of "off" shall be assigned to all modes not explicitly set to "on" in the PTYPE field, and all modes shall continue to have an inferred state of "off" until a new picture containing the optional part of PLUSPTYPE is sent.

4) Two modes do not require mode state inference, since the mode flags for these modes appear in the mandatory part of PLUSPTYPE. These are the Reference Picture Resampling mode (Annex P) and the Reduced-Resolution Update mode (Annex Q). The mode flag for either of these modes shall not be set unless the current picture allows the use of the mode. For example, the Reduced-Resolution Update mode bit shall not be set in an INTRA picture.

#### 5.1.4.6 Mode interaction restrictions

Certain modes cannot be used in combination with certain other modes.

1) The Syntax-based Arithmetic Coding mode (see Annex E) shall not be used with the Alternative INTER VLC mode (see Annex S) or the Modified Quantization mode (see Annex T).

2) If PLUSPTYPE is present, the Unrestricted Motion Vector mode (see Annex D) shall not be used with the Syntax-based Arithmetic Coding mode (see Annex E).

3) The Independent Segment Decoding mode (see Annex R) shall not be used with the Reference Picture Resampling mode (see Annex P).

4) The Independent Segment Decoding mode (see Annex R) shall not be used with the Slice Structured mode without the simultaneous use of the Rectangular Slice submode of the Slice Structured mode (see Annex K).

#### 5.1.4.7 The picture header location of CPM (1 bit) and PSBI (2 bits)

The location of the CPM and PSBI fields in the picture header depends on whether or not PLUSPTYPE is present (see 5.1.20 and 5.1.21). If PLUSPTYPE is present, then CPM follows immediately after PLUSPTYPE in the picture header. If PLUSPTYPE is not present, then CPM follows immediately after PQUANT in the picture header. PSBI always follows immediately after CPM (if CPM = "1").

### 5.1.5 Custom Picture Format (CPFMT) (23 bits)

A fixed length codeword of 23 bits that is present only if the use of a custom picture format is signalled in PLUSPTYPE and UFEP is '001'. When present, CPFMT consists of:

– Bits 1-4 Pixel Aspect Ratio Code: A 4-bit index to the PAR value in Table 5. For extended PAR, the exact pixel aspect ratio shall be specified in EPAR (see 5.1.6);

– Bits 5-13 Picture Width Indication: Range [0, ..., 511]; Number of pixels per line = (PWI + 1) * 4;

– Bit 14 Equal to "1" to prevent start code emulation;

– Bits 15-23 Picture Height Indication: Range [1, ..., 288]; Number of lines = PHI * 4.

**Table 5/H.263 – PAR code definition**

| PAR code | Pixel aspect ratios |
|---|---|
| 0000 | Forbidden |
| 0001 | 1:1 (Square) |
| 0010 | 12:11 (CIF for 4:3 picture) |
| 0011 | 10:11 (525-type for 4:3 picture) |
| 0100 | 16:11 (CIF stretched for 16:9 picture) |
| 0101 | 40:33 (525-type stretched for 16:9 picture) |
| 0110-1110 | Reserved |
| 1111 | Extended PAR |

### 5.1.6 Extended Pixel Aspect Ratio (EPAR) (16 bits)

A fixed length codeword of 16 bits that is present only if CPFMT is present and extended PAR is indicated therein. When present, EPAR consists of:

– Bits 1-8 PAR Width: "0" is forbidden. The natural binary representation of the PAR width;

– Bits 9-16 PAR Height: "0" is forbidden. The natural binary representation of the PAR height.

The PAR Width and PAR Height shall be relatively prime.

### 5.1.7 Custom Picture Clock Frequency Code (CPCFC) (8 bits)

A fixed length codeword of 8 bits that is present only if PLUSPTYPE is present and UFEP is 001 and a custom picture clock frequency is signalled in PLUSPTYPE. When present, CPCFC consists of:

– Bit 1 Clock Conversion Code: "0" indicates a clock conversion factor of 1000 and "1" indicates 1001;

– Bits 2-8 Clock Divisor: "0" is forbidden. The natural binary representation of the value of the clock divisor.

The custom picture clock frequency is given by 1 800 000/(clock divisor * clock conversion factor) Hz.

The temporal reference counter shall count in units of the inverse of the picture clock frequency, in seconds. When the PCF changes from that specified for the previous picture, the temporal reference for the current picture is measured in terms of the prior PCF, so that the new PCF takes effect only for the temporal reference interpretation of future pictures.

### 5.1.8 Extended Temporal Reference (ETR) (2 bits)

A fixed length codeword of 2 bits which is present only if a custom picture clock frequency is in use (regardless of the value of UFEP). It is the two MSBs of the 10-bit number defined in 5.1.2.

### 5.1.9 Unlimited Unrestricted Motion Vectors Indicator (UUI) (Variable length)

A variable length codeword of 1 or 2 bits that is present only if the optional Unrestricted Motion Vector mode is indicated in PLUSPTYPE and UFEP is 001. When UUI is present it indicates the effective limitation of the range of the motion vectors being used.

– UUI = "1" The motion vector range is limited according to Tables D.1 and D.2.

– UUI = "01" The motion vector range is not limited except by the picture size.

### 5.1.10 Slice Structured Submode bits (SSS) (2 bits)

A fixed length codeword of 2 bits which is present only if the optional Slice Structured mode (see Annex K) is indicated in PLUSPTYPE and UFEP is 001. If the Slice Structured mode is in use but UFEP is not 001, the last values sent for SSS shall remain in effect.

– Bit 1 Rectangular Slices, "0" indicates free-running slices, "1" indicates rectangular slices;

– Bit 2 Arbitrary Slice Ordering, "0" indicates sequential order, "1" indicates arbitrary order.

### 5.1.11 Enhancement Layer Number (ELNUM) (4 bits)

A fixed length codeword of 4 bits which is present only if the optional Temporal, SNR, and Spatial Scalability mode is in use (regardless of the value of UFEP). The particular enhancement layer is identified by an enhancement layer number, ELNUM. Picture correspondence between layers is achieved via the temporal reference. Picture size is either indicated within each enhancement layer using the existing source format fields or is inferred by the relationship to the reference layer. The first enhancement layer above the base layer is designated as Enhancement Layer Number 2, and the base layer has number 1.

### 5.1.12 Reference Layer Number (RLNUM) (4 bits)

A fixed length codeword of 4 bits which is present only if the optional Temporal, SNR, and Spatial Scalability mode is in use (see Annex O) and UFEP is 001. The layer number for the pictures used as reference anchors is identified by a Reference Layer Number (RLNUM). Time correspondence between layers is achieved via the temporal reference.

Note that for B-pictures in an enhancement layer having temporally surrounding EI- or EP-pictures which are present in the same enhancement layer, RLNUM shall be equal to ELNUM (see Annex O).

### 5.1.13 Reference Picture Selection Mode Flags (RPSMF) (3 bits)

A fixed length codeword of 3 bits that is present only if the Reference Picture Selection mode is in use and UFEP is 001. When present, RPSMF indicates which type of back-channel messages are needed by the encoder. If the Reference Picture Selection mode is in use but RPSMF is not present, the last value of RPSMF that was sent shall remain in effect.

– 100: neither ACK nor NACK signals needed;

– 101: need ACK signals to be returned;

– 110: need NACK signals to be returned;

– 111: need both ACK and NACK signals to be returned;

– 000-011: Reserved.

### 5.1.14 Temporal Reference for Prediction Indication (TRPI) (1 bit)

A fixed length codeword of 1 bit that is present only if the optional Reference Picture Selection mode is in use (regardless of the value of UFEP). When present, TRPI indicates the presence of the following TRP field:

– 0: TRP field is not present;

– 1: TRP field is present.

TRPI shall be 0 whenever the picture header indicates an I- or EI-picture.

### 5.1.15 Temporal Reference for Prediction (TRP) (10 bits)

When present (as indicated in TRPI), TRP indicates the Temporal Reference which is used for prediction of the encoding, except for in the case of B-pictures. For B-pictures, the picture having the temporal reference TRP is used for the prediction in the forward direction. (Prediction in the reverse-temporal direction always uses the immediately temporally subsequent picture.) TRP is a ten-bit number. If a custom picture clock frequency was not in use for the reference picture, the two MSBs of TRP are zero and the LSBs contain the eight-bit TR found in the picture header of the reference picture. If a custom picture clock frequency was in use for the reference picture, TRP is a ten-bit number consisting of the concatenation of ETR and TR from the reference picture header.

When TRP is not present, the most recent temporally previous anchor picture shall be used for prediction, as when not in the Reference Picture Selection mode. TRP is valid until the next PSC, GSC, or SSC.

### 5.1.16 Back-Channel message Indication (BCI) (Variable length)

A variable length field of one or two bits that is present only if the optional Reference Picture Selection mode is in use. When set to "1", this signals the presence of the following optional video Back-Channel Message (BCM) field. "01" indicates the absence or the end of the video back-channel message field. Combinations of BCM and BCI may not be present, and may be repeated when present. BCI shall be set to "01" if the videomux submode of the optional Reference Picture Selection mode is not in use.

### 5.1.17 Back-Channel Message (BCM) (Variable length)

The Back-Channel message with syntax as specified in N.4.2, which is present only if the preceding BCI field is present and is set to "1".

### 5.1.18 Reference Picture Resampling Parameters (RPRP) (Variable length)

A variable length field that is present only if the optional Reference Picture Resampling mode bit is set in PLUSPTYPE. This field carries the parameters of the Reference Picture Resampling mode (see Annex P). Note that the Reference Picture Resampling mode can also be invoked implicitly by the occurrence of a picture header for an INTER coded picture having a picture size which differs from that of the previous encoded picture, in which case the RPRP field is not present and the Reference Picture Resampling mode bit is not set.

### 5.1.19 Quantizer Information (PQUANT) (5 bits)

A fixed length codeword of 5 bits which indicates the quantizer QUANT to be used for the picture until updated by any subsequent GQUANT or DQUANT. The codewords are the natural binary representations of the values of QUANT which, being half the step sizes, range from 1 to 31.

### 5.1.20 Continuous Presence Multipoint and Video Multiplex (CPM) (1 bit)

A codeword of 1 bit that signals the use of the optional Continuous Presence Multipoint and Video Multiplex mode (CPM); "0" is off, "1" is on. For the use of CPM, refer to Annex C. CPM follows immediately after PLUSPTYPE if PLUSPTYPE is present, but follows PQUANT in the picture header if PLUSPTYPE is not present.

### 5.1.21 Picture Sub-Bitstream Indicator (PSBI) (2 bits)

A fixed length codeword of 2 bits that is only present if Continuous Presence Multipoint and Video Multiplex mode is indicated by CPM. The codewords are the natural binary representation of the sub-bitstream number for the picture header and all following information until the next Picture or GOB start code (see also Annex C). PSBI follows immediately after CPM if CPM is "1" (the location of CPM and PSBI in the picture header depend on whether PLUSPTYPE is present).

### 5.1.22 Temporal Reference for B-pictures in PB-frames ($TR_B$) (3/5 bits)

$TR_B$ is present if PTYPE or PLUSPTYPE indicates "PB-frame" or "Improved PB-frame" (see also Annexes G and M) and indicates the number of non-transmitted or non-reference pictures (at 29.97 Hz or the custom picture clock frequency indicated in CPCFC) since the last P- or I-picture or P-part of a PB-or Improved PB-frame and before the B-picture part of the PB- or Improved PB-frame. The codeword is the natural binary representation of the number of non-transmitted pictures plus one. It is 3 bits long for standard CIF picture clock frequency and is extended to 5 bits when a custom picture clock frequency is in use. The maximum number of non-transmitted pictures is 6 for the standard CIF picture clock frequency and 30 when a custom picture clock frequency is used.

### 5.1.23 Quantization information for B-pictures in PB-frames (DBQUANT) (2 bits)

DBQUANT is present if PTYPE or PLUSPTYPE indicates "PB-frame" or "Improved PB-frame" (see also Annexes G and M). In the decoding process a quantization parameter QUANT is obtained for each macroblock. With PB-frames QUANT is used for the P-block, while for the B-block a different quantization parameter BQUANT is used. QUANT ranges from 1 to 31. DBQUANT indicates the relation between QUANT and BQUANT as defined in Table 6. In this table, "/" means division by truncation. BQUANT ranges from 1 to 31; if the value for BQUANT resulting from Table 6 is greater than 31, it is clipped to 31.

**Table 6/H.263 – DBQUANT codes and relation between QUANT and BQUANT**

| DBQUANT | BQUANT |
|---|---|
| 00 | (5 × QUANT)/4 |
| 01 | (6 × QUANT)/4 |
| 10 | (7 × QUANT)/4 |
| 11 | (8 × QUANT)/4 |

### 5.1.24 Extra Insertion Information (PEI) (1 bit)

A bit which when set to "1" signals the presence of the following optional data field.

### 5.1.25 Supplemental Enhancement Information (PSUPP) (0/8/16 ... bits)

If PEI is set to "1", then 9 bits follow consisting of 8 bits of data (PSUPP) and then another PEI bit to indicate if a further 9 bits follow and so on. Encoders shall use PSUPP as specified in Annex L.

Decoders which do not support the extended capabilities described in Annex L shall be designed to discard PSUPP if PEI is set to 1. This enables backward compatibility for the extended capabilities of Annex L so that a bitstream which makes use of the extended capabilities can also be used without alteration by decoders which do not support those capabilities.

### 5.1.26 Stuffing (ESTUF) (Variable length)

A codeword of variable length consisting of less than 8 zero-bits. Encoders may insert this codeword directly before an EOS codeword. Encoders shall insert this codeword as necessary to attain mandatory byte alignment directly before an EOSBS codeword. If ESTUF is present, the last bit of ESTUF shall be the last (least significant) bit of a byte, so that the start of the EOS or EOSBS codeword is byte aligned. Decoders shall be designed to discard ESTUF. See Annex C for a description of EOSBS and its use.

### 5.1.27 End Of Sequence (EOS) (22 bits)

A codeword of 22 bits. Its value is 0000 0000 0000 0000 1 11111. It is up to the encoder to insert this codeword or not. EOS may be byte aligned. This can be achieved by inserting ESTUF before the EOS code such that the first bit of the EOS code is the first (most significant) bit of a byte. EOS shall not be repeated unless at least one picture start code appears between each pair of EOS codes.

### 5.1.28 Stuffing (PSTUF) (Variable length)

A codeword of variable length consisting of less than 8 zero-bits. Encoders shall insert this codeword for byte alignment of the next PSC. The last bit of PSTUF shall be the last (least significant) bit of a byte, so that the video bitstream including PSTUF is a multiple of 8 bits from the first bit in the H.263 bitstream. Decoders shall be designed to discard PSTUF.

If for some reason the encoder stops encoding pictures for a certain time-period and resumes encoding later, PSTUF shall be transmitted before the encoder stops, to prevent that the last up to 7 bits of the previous picture are not sent until the coder resumes coding.

## 5.2 Group of Blocks Layer

Data for each Group of Blocks (GOB) consists of a GOB header followed by data for macroblocks. The structure is shown in Figure 9. Each GOB contains one or more rows of macroblocks. For the first GOB in each picture (with number 0), no GOB header shall be transmitted. For all other GOBs, the GOB header may be empty, depending on the encoder strategy. A decoder can signal the remote encoder to transmit only non-empty GOB headers by external means, for example Recommendation H.245. GSTUF may be present when GBSC is present. GN, GFID and GQUANT are present when GBSC is present. GSBI is present when CPM is "1" in the Picture header.

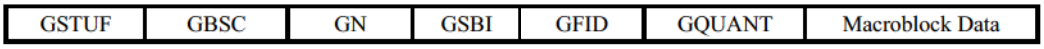



**Figure 9/H.263 – Structure of GOB layer**

### 5.2.1 Stuffing (GSTUF) (Variable length)

A codeword of variable length consisting of less than 8 zero-bits. Encoders may insert this codeword directly before a GBSC codeword. If GSTUF is present, the last bit of GSTUF shall be the last (least significant) bit of a byte, so that the start of the GBSC codeword is byte aligned. Decoders shall be designed to discard GSTUF.

### 5.2.2 Group of Block Start Code (GBSC) (17 bits)

A word of 17 bits. Its value is 0000 0000 0000 0000 1. GOB start codes may be byte aligned. This can be achieved by inserting GSTUF before the start code such that the first bit of the start code is the first (most significant) bit of a byte.

### 5.2.3 Group Number (GN) (5 bits)

A fixed length codeword of 5 bits. The bits are the binary representation of the number of the Group of Blocks. For the GOB with number 0, the GOB header including GSTUF, GBSC, GN, GSBI, GFID and GQUANT is empty; as group number 0 is used in the PSC. Group numbers 1 through 17 are used in GOB headers of the standard picture formats. Group numbers 1 through 24 are used in GOB headers of custom picture formats. Group numbers 16 through 28 are emulated in the slice header (see Annex K) when CPM = "0", and Group Numbers 25-27 and 29 are emulated in the slice header (see Annex K) when CPM = "1". Group number 31 is used in the EOS code, and group number 30 is used in the EOSBS code.

### 5.2.4 GOB Sub-Bitstream Indicator (GSBI) (2 bits)

A fixed length codeword of 2 bits that is only present if CPM is "1" in the picture header. The codewords are the natural binary representation of the sub-bitstream number for the GOB header and all following information until the next Picture or GOB start code (see also Annex C).

### 5.2.5 GOB Frame ID (GFID) (2 bits)

A fixed length codeword of 2 bits. GFID shall have the same value in every GOB (or slice) header of a given picture. Moreover, if PTYPE as indicated in a picture header is the same as for the previous transmitted picture, GFID shall have the same value as in that previous picture, provided PLUSPTYPE is not present. However, if PTYPE in a certain picture header differs from the PTYPE in the previous transmitted picture header, the value for GFID in that picture shall differ from the value in the previous picture.

If PLUSPTYPE is present, the value of GFID shall be the same as that for the previous picture (in the same layer) if the PTYPE, and PLUSPTYPE, and all of the present fields among CPFMT, EPAR, CPCFC, SSS, ELNUM, RLNUM, UUI, RPSMF, and RPRP remain in effect as for the previous picture; otherwise, GFID shall be different from that for the previous picture.

### 5.2.6 Quantizer Information (GQUANT) (5 bits)

A fixed length codeword of 5 bits which indicates the quantizer QUANT to be used for the remaining part of the picture until updated by any subsequent GQUANT or DQUANT. The codewords are the natural binary representations of the values of QUANT which, being half the step sizes, range from 1 to 31.

## 5.3 Macroblock layer

Data for each macroblock consists of a macroblock header followed by data for blocks. The structure is shown in Figure 10. COD is only present in pictures that are not of type 'INTRA', for each macroblock in these pictures. MCBPC is present when indicated by COD or when the picture is of type 'INTRA'. MODB is present for MB-type 0-4 if PTYPE indicates "PB-frame". CBPY, DQUANT, MVD and $MVD_{2\text{-}4}$ are present when indicated by MCBPC. CBPB and MVDB are only present if indicated by MODB. Block Data is present when indicated by MCBPC and CBPY. $MVD_{2\text{-}4}$ are only present when in Advanced Prediction mode (refer to Annex F) or Deblocking Filter mode (refer to Annex J). MODB, CBPB and MVDB are only present in PB-frames mode (refer to Annex G). For coding of the symbols in the Syntax-based Arithmetic Coding mode, refer to Annex E. For coding of the macroblock layer in B-, EI-, and EP-pictures, see Annex O.

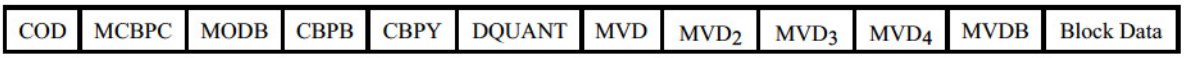



**Figure 10/H.263 – Structure of macroblock layer**

### 5.3.1 Coded macroblock indication (COD) (1 bit)

A bit which when set to "0" signals that the macroblock is coded. If set to "1", no further information is transmitted for this macroblock; in that case the decoder shall treat the macroblock as an INTER macroblock with motion vector for the whole block equal to zero and with no coefficient data. COD is only present in pictures that are not of type "INTRA", for each macroblock in these pictures.

NOTE – In Advanced Prediction mode (see Annex F), overlapped block motion compensation is also performed if COD is set to "1"; and in Deblocking Filter mode (see Annex J), the deblocking filter can also affect the values of some pixels of macroblocks having COD is set to "1".

### 5.3.2 Macroblock type & Coded Block Pattern for Chrominance (MCBPC) (Variable length)

MCBPC is a variable length codeword giving information about the macroblock type and the coded block pattern for chrominance. The codewords for MCBPC are given in Tables 7 and 8. MCBPC is always included in coded macroblocks.

**Table 7/H.263 – VLC table for MCBPC (for I-pictures)**

| Index | MB type | CBPC (56) | Number of bits | Code |
|---|---|---|---|---|
| 0 | 3 | 00 | 1 | 1 |
| 1 | 3 | 01 | 3 | 001 |
| 2 | 3 | 10 | 3 | 010 |
| 3 | 3 | 11 | 3 | 011 |
| 4 | 4 | 00 | 4 | 0001 |
| 5 | 4 | 01 | 6 | 0000 01 |
| 6 | 4 | 10 | 6 | 0000 10 |
| 7 | 4 | 11 | 6 | 0000 11 |
| 8 | Stuffing | – | 9 | 0000 0000 1 |

**Table 8/H.263 – VLC table for MCBPC (for P-pictures)**

| Index | MB type | CBPC (56) | Number of bits | Code |
|---|---|---|---|---|
| 0 | 0 | 00 | 1 | 1 |
| 1 | 0 | 01 | 4 | 0011 |
| 2 | 0 | 10 | 4 | 0010 |
| 3 | 0 | 11 | 6 | 0001 01 |
| 4 | 1 | 00 | 3 | 011 |
| 5 | 1 | 01 | 7 | 0000 111 |
| 6 | 1 | 10 | 7 | 0000 110 |
| 7 | 1 | 11 | 9 | 0000 0010 1 |
| 8 | 2 | 00 | 3 | 010 |
| 9 | 2 | 01 | 7 | 0000 101 |
| 10 | 2 | 10 | 7 | 0000 100 |
| 11 | 2 | 11 | 8 | 0000 0101 |
| 12 | 3 | 00 | 5 | 0001 1 |
| 13 | 3 | 01 | 8 | 0000 0100 |
| 14 | 3 | 10 | 8 | 0000 0011 |
| 15 | 3 | 11 | 7 | 0000 011 |
| 16 | 4 | 00 | 6 | 0001 00 |
| 17 | 4 | 01 | 9 | 0000 0010 0 |
| 18 | 4 | 10 | 9 | 0000 0001 1 |
| 19 | 4 | 11 | 9 | 0000 0001 0 |
| 20 | Stuffing | – | 9 | 0000 0000 1 |
| 21 | 5 | 00 | 11 | 0000 0000 010 |
| 22 | 5 | 01 | 13 | 0000 0000 0110 0 |
| 23 | 5 | 10 | 13 | 0000 0000 0111 0 |
| 24 | 5 | 11 | 13 | 0000 0000 0111 1 |

An extra codeword is available in the tables for bit stuffing. This codeword should be discarded by decoders. If an Improved PB-frame is indicated by MPPTYPE bits 1-3 and Custom Source Format is indicated in OPPTYPE bits 1-3, then MBA shall not indicate stuffing before the first macroblock of the picture (in order to prevent start code emulation).

The macroblock type gives information about the macroblock and which data elements are present. Macroblock types and included elements are listed in Tables 9 and 10. Macroblock type 5 (indices 21-24 in Table 8) shall not be present unless an extended PTYPE (PLUSPTYPE) is present in the picture header and either the Advanced Prediction mode (see Annex F) or the Deblocking Filter mode (see Annex J) is in use, and shall not be present for the first macroblock of a picture. Also, encoders shall not allow an MCBPC code for macroblock type 5 to immediately follow seven consecutive zeros in the bitstream (as can be caused by particular INTRADC codes followed by COD = 0), in order to prevent start code emulation. Codes for macroblock type 5 can be preceded by stuffing when necessary to fulfill this requirement (for macroblocks other than the first of a picture).

**Table 9/H.263 – Macroblock types and included data elements for normal pictures**

| Picture type | MB type | Name | COD | MCBPC | CBPY | DQUANT | MVD | $MVD_{2-4}$ |
|---|---|---|---|---|---|---|---|---|
| INTER | Not coded | – | X | | | | | |
| INTER | 0 | INTER | X | X | X | | X | |
| INTER | 1 | INTER+Q | X | X | X | X | X | |
| INTER | 2 | INTER4V | X | X | X | | X | X |
| INTER | 3 | INTRA | X | X | X | | | |
| INTER | 4 | INTRA+Q | X | X | X | X | | |
| INTER | 5 | INTER4V+Q | X | X | X | X | X | X |
| INTER | Stuffing | – | X | X | | | | |
| INTRA | 3 | INTRA | | X | X | | | |
| INTRA | 4 | INTRA+Q | | X | X | X | | |
| INTRA | Stuffing | – | | X | | | | |
| NOTE – "X" means that the item is present in the macroblock. | | | | | | | | |

**Table 10/H.263 – Macroblock types and included data elements for PB-frames**

| Picture type | MB type | Name | COD | MCBPC | MOD B | CBPY | CBP B | DQUANT | MVD | MVDB | $MVD_{2-4}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| INTER | Not coded | – | X | | | | | | | | |
| INTER | 0 | INTER | X | X | X | X | (X) | | X | (X) | |
| INTER | 1 | INTER+Q | X | X | X | X | (X) | X | X | (X) | |
| INTER | 2 | INTER4V | X | X | X | X | (X) | | X | (X) | X |
| INTER | 3 | INTRA | X | X | X | X | (X) | | X | (X) | |
| INTER | 4 | INTRA+Q | X | X | X | X | (X) | X | X | (X) | |
| INTER | 5 | INTER4V+Q | X | X | X | X | (X) | X | X | (X) | X |
| INTER | Stuffing | – | X | X | | | | | | | |
| NOTE 1 – "X" means that the item is present in the macroblock.<br>NOTE 2 – CBPB and MVDB are only present if indicated by MODB.<br>NOTE 3 – B-blocks are always coded in INTER mode, even if the macroblock type of the PB-macroblock indicates INTRA. | | | | | | | | | | | |

The coded block pattern for chrominance signifies $C_B$ and/or $C_R$ blocks when at least one non-INTRADC transform coefficient is transmitted (INTRADC is the dc-coefficient for INTRA blocks, see 5.4.1), unless the optional Advanced INTRA Coding mode is in use. $CBPC_N = 1$ if any non-INTRADC coefficient is present for block N, else 0, for $CBPC_5$ and $CBPC_6$ in the coded block pattern. If Advanced INTRA Coding is in use, the usage is similar, but the INTRADC coefficient is indicated in the same way as the other coefficients (see Annex I). Block numbering is given in Figure 5. When MCBPC = Stuffing, the remaining part of the macroblock layer is skipped. In this case, the preceding COD = 0 is not related to any coded or not-coded macroblock and therefore the macroblock number is not incremented. For P-pictures, multiple stuffings are accomplished by multiple sets of COD = 0 and MCBPC = Stuffing. See Tables 7 and 8.

### 5.3.3 Macroblock mode for B-blocks (MODB) (Variable length)

MODB is present for MB-type 0-4 if PTYPE indicates "PB-frame" and is a variable length codeword indicating whether CBPB is present (indicates that B-coefficients are transmitted for this macroblock) and/or MVDB is present. In Table 11 the codewords for MODB are defined. MODB is coded differently for Improved PB-frames, as specified in Annex M.

**Table 11/H.263 – VLC table for MODB**

| Index | CBPB | MVDB | Number of bits | Code |
|---|---|---|---|---|
| 0 | | | 1 | 0 |
| 1 | | X | 2 | 10 |
| 2 | X | X | 2 | 11 |
| NOTE – "X" means that the item is present in the macroblock. | | | | |

### 5.3.4 Coded Block Pattern for B-blocks (CBPB) (6 bits)

CBPB is only present in PB-frames mode if indicated by MODB. $CBPB_N$ = 1 if any coefficient is present for B-block N, else 0, for each bit $CBPB_N$ in the coded block pattern. Block numbering is given in Figure 5, the utmost left bit of CBPB corresponding with block number 1.

### 5.3.5 Coded Block Pattern for luminance (CBPY) (Variable length)

Variable length codeword giving a pattern number signifying those Y blocks in the macroblock for which at least one non-INTRADC transform coefficient is transmitted (INTRADC is the dc-coefficient for INTRA blocks, see 5.4.1), unless the Advanced INTRA Coding mode is in use. If Advanced INTRA Coding is in use, INTRADC is indicated in the same manner as the other coefficients (see Annex I).

$CBPY_N$ = 1 if any non-INTRADC coefficient is present for block N, else 0, for each bit $CBPY_N$ in the coded block pattern. Block numbering is given in Figure 5, the utmost left bit of CBPY corresponding with block number 1. For a certain pattern $CBPY_N$, different codewords are used for INTER and INTRA macroblocks as defined in Table 13.

### 5.3.6 Quantizer Information (DQUANT) (2 bits/Variable Length)

If the Modified Quantization mode is not in use, DQUANT is a two-bit code to define a change in QUANT. In Table 12 the differential values for the different codewords are given. QUANT ranges from 1 to 31; if the value for QUANT after adding the differential value is less than 1 or greater than 31, it is clipped to 1 and 31 respectively. If the Modified Quantization mode is in use, DQUANT is a variable length codeword as specified in Annex T.

**Table 12/H.263 – DQUANT codes and differential values for QUANT**

| Index | Differential value | DQUANT |
|---|---|---|
| 0 | –1 | 00 |
| 1 | –2 | 01 |
| 2 | 1 | 10 |
| 3 | 2 | 11 |

**Table 13/H.263 – VLC table for CBPY**

| Index | CBPY(INTRA) (12, 34) | CBPY(INTER) (12, 34) | Number of bits | Code |
|---|---|---|---|---|
| 0 | 00<br>00 | 11<br>11 | 4 | 0011 |
| 1 | 00<br>01 | 11<br>10 | 5 | 0010 1 |
| 2 | 00<br>10 | 11<br>01 | 5 | 0010 0 |
| 3 | 00<br>11 | 11<br>00 | 4 | 1001 |
| 4 | 01<br>00 | 10<br>11 | 5 | 0001 1 |
| 5 | 01<br>01 | 10<br>10 | 4 | 0111 |
| 6 | 01<br>10 | 10<br>01 | 6 | 0000 10 |
| 7 | 01<br>11 | 10<br>00 | 4 | 1011 |
| 8 | 10<br>00 | 01<br>11 | 5 | 0001 0 |
| 9 | 10<br>01 | 01<br>10 | 6 | 0000 11 |
| 10 | 10<br>10 | 01<br>01 | 4 | 0101 |
| 11 | 10<br>11 | 01<br>00 | 4 | 1010 |
| 12 | 11<br>00 | 00<br>11 | 4 | 0100 |
| 13 | 11<br>01 | 00<br>10 | 4 | 1000 |
| 14 | 11<br>10 | 00<br>01 | 4 | 0110 |
| 15 | 11<br>11 | 00<br>00 | 2 | 11 |

### 5.3.7 Motion Vector Data (MVD) (Variable length)

MVD is included for all INTER macroblocks (in PB-frames mode also for INTRA macroblocks) and consists of a variable length codeword for the horizontal component followed by a variable length codeword for the vertical component. Variable length codes are given in Table 14. If the Unrestricted Motion Vector mode is used and PLUSPTYPE is present, motion vectors are coded using Table D.3 instead of Table 14 (see Annex D).

**Table 14/H.263 – VLC table for MVD**

| Index | Vector | Differences | Bit number | Codes |
|---|---|---|---|---|
| 0 | –16 | 16 | 13 | 0000 0000 0010 1 |
| 1 | –15.5 | 16.5 | 13 | 0000 0000 0011 1 |
| 2 | –15 | 17 | 12 | 0000 0000 0101 |
| 3 | –14.5 | 17.5 | 12 | 0000 0000 0111 |
| 4 | –14 | 18 | 12 | 0000 0000 1001 |
| 5 | –13.5 | 18.5 | 12 | 0000 0000 1011 |
| 6 | –13 | 19 | 12 | 0000 0000 1101 |
| 7 | –12.5 | 19.5 | 12 | 0000 0000 1111 |
| 8 | –12 | 20 | 11 | 0000 0001 001 |
| 9 | –11.5 | 20.5 | 11 | 0000 0001 011 |
| 10 | –11 | 21 | 11 | 0000 0001 101 |
| 11 | –10.5 | 21.5 | 11 | 0000 0001 111 |
| 12 | –10 | 22 | 11 | 0000 0010 001 |
| 13 | –9.5 | 22.5 | 11 | 0000 0010 011 |
| 14 | –9 | 23 | 11 | 0000 0010 101 |
| 15 | –8.5 | 23.5 | 11 | 0000 0010 111 |
| 16 | –8 | 24 | 11 | 0000 0011 001 |
| 17 | –7.5 | 24.5 | 11 | 0000 0011 011 |
| 18 | –7 | 25 | 11 | 0000 0011 101 |
| 19 | –6.5 | 25.5 | 11 | 0000 0011 111 |
| 20 | –6 | 26 | 11 | 0000 0100 001 |
| 21 | –5.5 | 26.5 | 11 | 0000 0100 011 |
| 22 | –5 | 27 | 10 | 0000 0100 11 |
| 23 | –4.5 | 27.5 | 10 | 0000 0101 01 |
| 24 | –4 | 28 | 10 | 0000 0101 11 |
| 25 | –3.5 | 28.5 | 8 | 0000 0111 |
| 26 | –3 | 29 | 8 | 0000 1001 |
| 27 | –2.5 | 29.5 | 8 | 0000 1011 |
| 28 | –2 | 30 | 7 | 0000 111 |
| 29 | –1.5 | 30.5 | 5 | 0001 1 |
| 30 | –1 | 31 | 4 | 0011 |
| 31 | –0.5 | 31.5 | 3 | 011 |
| 32 | 0 | | 1 | 1 |
| 33 | 0.5 | –31.5 | 3 | 010 |
| 34 | 1 | –31 | 4 | 0010 |
| 35 | 1.5 | –30.5 | 5 | 0001 0 |
| 36 | 2 | –30 | 7 | 0000 110 |
| 37 | 2.5 | –29.5 | 8 | 0000 1010 |
| 38 | 3 | –29 | 8 | 0000 1000 |
| 39 | 3.5 | –28.5 | 8 | 0000 0110 |
| 40 | 4 | –28 | 10 | 0000 0101 10 |
| 41 | 4.5 | –27.5 | 10 | 0000 0101 00 |
| 42 | 5 | –27 | 10 | 0000 0100 10 |
| 43 | 5.5 | –26.5 | 11 | 0000 0100 010 |
| 44 | 6 | –26 | 11 | 0000 0100 000 |
| 45 | 6.5 | –25.5 | 11 | 0000 0011 110 |

**Table 14/H.263 – VLC table for MVD** (*concluded*)

| Index | Vector | Differences | Bit number | Codes |
|---|---|---|---|---|
| 46 | 7 | –25 | 11 | 0000 0011 100 |
| 47 | 7.5 | –24.5 | 11 | 0000 0011 010 |
| 48 | 8 | –24 | 11 | 0000 0011 000 |
| 49 | 8.5 | –23.5 | 11 | 0000 0010 110 |
| 50 | 9 | –23 | 11 | 0000 0010 100 |
| 51 | 9.5 | –22.5 | 11 | 0000 0010 010 |
| 52 | 10 | –22 | 11 | 0000 0010 000 |
| 53 | 10.5 | –21.5 | 11 | 0000 0001 110 |
| 54 | 11 | –21 | 11 | 0000 0001 100 |
| 55 | 11.5 | –20.5 | 11 | 0000 0001 010 |
| 56 | 12 | –20 | 11 | 0000 0001 000 |
| 57 | 12.5 | –19.5 | 12 | 0000 0000 1110 |
| 58 | 13 | –19 | 12 | 0000 0000 1100 |
| 59 | 13.5 | –18.5 | 12 | 0000 0000 1010 |
| 60 | 14 | –18 | 12 | 0000 0000 1000 |
| 61 | 14.5 | –17.5 | 12 | 0000 0000 0110 |
| 62 | 15 | –17 | 12 | 0000 0000 0100 |
| 63 | 15.5 | –16.5 | 13 | 0000 0000 0011 0 |

### 5.3.8 Motion Vector Data ($MVD_{2\text{-}4}$) (Variable length)

The three codewords $MVD_{2\text{-}4}$ are included if indicated by PTYPE and by MCBPC, and consist each of a variable length codeword for the horizontal component followed by a variable length codeword for the vertical component of each vector. Variable length codes are given in Table 14. $MVD_{2\text{-}4}$ are only present when in Advanced Prediction mode (see Annex F) or Deblocking Filter mode (see Annex J).

### 5.3.9 Motion Vector Data for B-macroblock (MVDB) (Variable length)

MVDB is only present in PB-frames or Improved PB-frames mode if indicated by MODB, and consists of a variable length codeword for the horizontal component followed by a variable length codeword for the vertical component of each vector. Variable length codes are given in Table 14. For the use of MVDB, refer to Annexes G and M.

## 5.4 Block layer

If not in PB-frames mode, a macroblock comprises four luminance blocks and one of each of the two colour difference blocks (see Figure 5). The structure of the block layer is shown in Figure 11. INTRADC is present for every block of the macroblock if MCBPC indicates macroblock type 3 or 4 (see Tables 7 and 8). TCOEF is present if indicated by MCBPC or CBPY.

In PB-frames mode, a macroblock comprises twelve blocks. First the data for the six P-blocks is transmitted as in the default H.263 mode, then the data for the six B-blocks. INTRADC is present for every P-block of the macroblock if MCBPC indicates macroblock type 3 or 4 (see Tables 7 and 8). INTRADC is not present for B-blocks. TCOEF is present for P-blocks if indicated by MCBPC or CBPY. TCOEF is present for B-blocks if indicated by CBPB.

For coding of the symbols in the Syntax-based Arithmetic Coding mode, refer to Annex E.

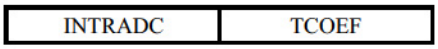



**Figure 11/H.263 – Structure of block layer**

### 5.4.1 DC coefficient for INTRA blocks (INTRADC) (8 bits)

A codeword of 8 bits. The code 0000 0000 is not used. The code 1000 0000 is not used, the reconstruction level of 1024 being coded as 1111 1111 (see Table 15).

**Table 15/H.263 – Reconstruction levels for INTRA-mode DC coefficient**

| Index | FLC | Reconstruction level into inverse transform |
|---|---|---|
| 0 | 0000 0001 (1) | 8 |
| 1 | 0000 0010 (2) | 16 |
| 2 | 0000 0011 (3) | 24 |
| . | . . | . |
| . | . . | . |
| 126 | 0111 1111 (127) | 1016 |
| 127 | 1111 1111 (255) | 1024 |
| 128 | 1000 0001 (129) | 1032 |
| . | . . | . |
| . | . . | . |
| 252 | 1111 1101 (253) | 2024 |
| 253 | 1111 1110 (254) | 2032 |

### 5.4.2 Transform Coefficient (TCOEF) (Variable length)

The most commonly occurring EVENTs are coded with the variable length codes given in Table 16. The last bit "s" denotes the sign of the level, "0" for positive and "1" for negative.

An EVENT is a combination of a last non-zero coefficient indication (LAST; "0": there are more non-zero coefficients in this block, "1": this is the last non-zero coefficient in this block), the number of successive zeros preceding the coded coefficient (RUN), and the non-zero value of the coded coefficient (LEVEL).

The remaining combinations of (LAST, RUN, LEVEL) are coded with a 22-bit word consisting of 7 bits ESCAPE, 1 bit LAST, 6 bits RUN and 8 bits LEVEL. Use of this 22-bit word for encoding the combinations listed in Table 16 is not prohibited. For the 8-bit word for LEVEL, the code 0000 0000 is forbidden, and the code 1000 0000 is forbidden unless the Modified Quantization mode is in use (see Annex T). The codes for RUN and for LEVEL are given in Table 17.

**Table 16/H.263 – VLC table for TCOEF**

| INDEX | LAST | RUN | \|LEVEL\| | BITS | VLC CODE |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 3 | 10s |
| 1 | 0 | 0 | 2 | 5 | 1111s |
| 2 | 0 | 0 | 3 | 7 | 0101 01s |
| 3 | 0 | 0 | 4 | 8 | 0010 111s |
| 4 | 0 | 0 | 5 | 9 | 0001 1111s |
| 5 | 0 | 0 | 6 | 10 | 0001 0010 1s |
| 6 | 0 | 0 | 7 | 10 | 0001 0010 0s |
| 7 | 0 | 0 | 8 | 11 | 0000 1000 01s |
| 8 | 0 | 0 | 9 | 11 | 0000 1000 00s |
| 9 | 0 | 0 | 10 | 12 | 0000 0000 111s |
| 10 | 0 | 0 | 11 | 12 | 0000 0000 110s |
| 11 | 0 | 0 | 12 | 12 | 0000 0100 000s |
| 12 | 0 | 1 | 1 | 4 | 110s |
| 13 | 0 | 1 | 2 | 7 | 0101 00s |
| 14 | 0 | 1 | 3 | 9 | 0001 1110s |
| 15 | 0 | 1 | 4 | 11 | 0000 0011 11s |
| 16 | 0 | 1 | 5 | 12 | 0000 0100 001s |
| 17 | 0 | 1 | 6 | 13 | 0000 0101 0000s |
| 18 | 0 | 2 | 1 | 5 | 1110s |
| 19 | 0 | 2 | 2 | 9 | 0001 1101s |
| 20 | 0 | 2 | 3 | 11 | 0000 0011 10s |
| 21 | 0 | 2 | 4 | 13 | 0000 0101 0001s |
| 22 | 0 | 3 | 1 | 6 | 0110 1s |
| 23 | 0 | 3 | 2 | 10 | 0001 0001 1s |
| 24 | 0 | 3 | 3 | 11 | 0000 0011 01s |
| 25 | 0 | 4 | 1 | 6 | 0110 0s |
| 26 | 0 | 4 | 2 | 10 | 0001 0001 0s |
| 27 | 0 | 4 | 3 | 13 | 0000 0101 0010s |
| 28 | 0 | 5 | 1 | 6 | 0101 1s |
| 29 | 0 | 5 | 2 | 11 | 0000 0011 00s |
| 30 | 0 | 5 | 3 | 13 | 0000 0101 0011s |
| 31 | 0 | 6 | 1 | 7 | 0100 11s |
| 32 | 0 | 6 | 2 | 11 | 0000 0010 11s |
| 33 | 0 | 6 | 3 | 13 | 0000 0101 0100s |
| 34 | 0 | 7 | 1 | 7 | 0100 10s |
| 35 | 0 | 7 | 2 | 11 | 0000 0010 10s |
| 36 | 0 | 8 | 1 | 7 | 0100 01s |
| 37 | 0 | 8 | 2 | 11 | 0000 0010 01s |
| 38 | 0 | 9 | 1 | 7 | 0100 00s |
| 39 | 0 | 9 | 2 | 11 | 0000 0010 00s |
| 40 | 0 | 10 | 1 | 8 | 0010 110s |
| 41 | 0 | 10 | 2 | 13 | 0000 0101 0101s |
| 42 | 0 | 11 | 1 | 8 | 0010 101s |
| 43 | 0 | 12 | 1 | 8 | 0010 100s |
| 44 | 0 | 13 | 1 | 9 | 0001 1100s |
| 45 | 0 | 14 | 1 | 9 | 0001 1011s |

**Table 16/H.263 – VLC table for TCOEF** (*continued*)

| INDEX | LAST | RUN | \|LEVEL\| | BITS | VLC CODE |
|---|---|---|---|---|---|
| 46 | 0 | 15 | 1 | 10 | 0001 0000 1s |
| 47 | 0 | 16 | 1 | 10 | 0001 0000 0s |
| 48 | 0 | 17 | 1 | 10 | 0000 1111 1s |
| 49 | 0 | 18 | 1 | 10 | 0000 1111 0s |
| 50 | 0 | 19 | 1 | 10 | 0000 1110 1s |
| 51 | 0 | 20 | 1 | 10 | 0000 1110 0s |
| 52 | 0 | 21 | 1 | 10 | 0000 1101 1s |
| 53 | 0 | 22 | 1 | 10 | 0000 1101 0s |
| 54 | 0 | 23 | 1 | 12 | 0000 0100 010s |
| 55 | 0 | 24 | 1 | 12 | 0000 0100 011s |
| 56 | 0 | 25 | 1 | 13 | 0000 0101 0110s |
| 57 | 0 | 26 | 1 | 13 | 0000 0101 0111s |
| 58 | 1 | 0 | 1 | 5 | 0111s |
| 59 | 1 | 0 | 2 | 10 | 0000 1100 1s |
| 60 | 1 | 0 | 3 | 12 | 0000 0000 101s |
| 61 | 1 | 1 | 1 | 7 | 0011 11s |
| 62 | 1 | 1 | 2 | 12 | 0000 0000 100s |
| 63 | 1 | 2 | 1 | 7 | 0011 10s |
| 64 | 1 | 3 | 1 | 7 | 0011 01s |
| 65 | 1 | 4 | 1 | 7 | 0011 00s |
| 66 | 1 | 5 | 1 | 8 | 0010 011s |
| 67 | 1 | 6 | 1 | 8 | 0010 010s |
| 68 | 1 | 7 | 1 | 8 | 0010 001s |
| 69 | 1 | 8 | 1 | 8 | 0010 000s |
| 70 | 1 | 9 | 1 | 9 | 0001 1010s |
| 71 | 1 | 10 | 1 | 9 | 0001 1001s |
| 72 | 1 | 11 | 1 | 9 | 0001 1000s |
| 73 | 1 | 12 | 1 | 9 | 0001 0111s |
| 74 | 1 | 13 | 1 | 9 | 0001 0110s |
| 75 | 1 | 14 | 1 | 9 | 0001 0101s |
| 76 | 1 | 15 | 1 | 9 | 0001 0100s |
| 77 | 1 | 16 | 1 | 9 | 0001 0011s |
| 78 | 1 | 17 | 1 | 10 | 0000 1100 0s |
| 79 | 1 | 18 | 1 | 10 | 0000 1011 1s |
| 80 | 1 | 19 | 1 | 10 | 0000 1011 0s |
| 81 | 1 | 20 | 1 | 10 | 0000 1010 1s |
| 82 | 1 | 21 | 1 | 10 | 0000 1010 0s |
| 83 | 1 | 22 | 1 | 10 | 0000 1001 1s |
| 84 | 1 | 23 | 1 | 10 | 0000 1001 0s |
| 85 | 1 | 24 | 1 | 10 | 0000 1000 1s |
| 86 | 1 | 25 | 1 | 11 | 0000 0001 11s |
| 87 | 1 | 26 | 1 | 11 | 0000 0001 10s |
| 88 | 1 | 27 | 1 | 11 | 0000 0001 01s |
| 89 | 1 | 28 | 1 | 11 | 0000 0001 00s |
| 90 | 1 | 29 | 1 | 12 | 0000 0100 100s |
| 91 | 1 | 30 | 1 | 12 | 0000 0100 101s |
| 92 | 1 | 31 | 1 | 12 | 0000 0100 110s |
| 93 | 1 | 32 | 1 | 12 | 0000 0100 111s |
| 94 | 1 | 33 | 1 | 13 | 0000 0101 1000s |

**Table 16/H.263 – VLC table for TCOEF** (*concluded*)

| INDEX | LAST | RUN | \|LEVEL\| | BITS | VLC CODE |
|---|---|---|---|---|---|
| 95 | 1 | 34 | 1 | 13 | 0000 0101 1001s |
| 96 | 1 | 35 | 1 | 13 | 0000 0101 1010s |
| 97 | 1 | 36 | 1 | 13 | 0000 0101 1011s |
| 98 | 1 | 37 | 1 | 13 | 0000 0101 1100s |
| 99 | 1 | 38 | 1 | 13 | 0000 0101 1101s |
| 100 | 1 | 39 | 1 | 13 | 0000 0101 1110s |
| 101 | 1 | 40 | 1 | 13 | 0000 0101 1111s |
| 102 | ESCAPE | | | 7 | 0000 011 |

**Table 17/H.263 – FLC table for RUNS and LEVELS**

| Index | Run | Code |
|---|---|---|
| 0 | 0 | 000 000 |
| 1 | 1 | 000 001 |
| 2 | 2 | 000 010 |
| . | . | . |
| . | . | . |
| 63 | 63 | 111 111 |

| Index | Level | Code |
|---|---|---|
| – | –128 | see text |
| 0 | –127 | 1000 0001 |
| . | . | . |
| 125 | –2 | 1111 1110 |
| 126 | –1 | 1111 1111 |
| – | 0 | FORBIDDEN |
| 127 | 1 | 0000 0001 |
| 128 | 2 | 0000 0010 |
| . | . | . |
| 253 | 127 | 0111 1111 |

# 6 Decoding process

## 6.1 Motion compensation

In this subclause, the motion compensation for the default H.263 prediction mode is described. For a description of motion compensation in the Unrestricted Motion Vector mode, refer to Annex D. For a description of motion compensation in the Advanced Prediction mode, refer to Annex F. For a description of motion compensation in the Reduced-Resolution Update mode refer to Annex Q.

### 6.1.1 Differential motion vectors

The macroblock vector is obtained by adding predictors to the vector differences indicated by MVD (see Table 14 and Table D.3). For differential coding with four vectors per macroblock, refer to Annex F. In case of one vector per macroblock, the candidate predictors for the differential coding are taken from three surrounding macroblocks as indicated in Figure 12. The predictors are calculated separately for the horizontal and vertical components.

In the special cases at the borders of the current GOB, slice, or picture, the following decision rules are applied in increasing order:

1) When the corresponding macroblock was coded in INTRA mode (if not in PB-frames mode with bidirectional prediction) or was not coded (COD = 1), the candidate predictor is set to zero.

2) The candidate predictor MV1 is set to zero if the corresponding macroblock is outside the picture or the slice (at the left side).

3) Then, the candidate predictors MV2 and MV3 are set to MV1 if the corresponding macroblocks are outside the picture (at the top) or outside the GOB (at the top) if the GOB header of the current GOB is non-empty; or outside the slice when in Slice Structured mode.

4) Then, the candidate predictor MV3 is set to zero if the corresponding macroblock is outside the picture (at the right side).

For each component, the predictor is the median value of the three candidate predictors for this component.

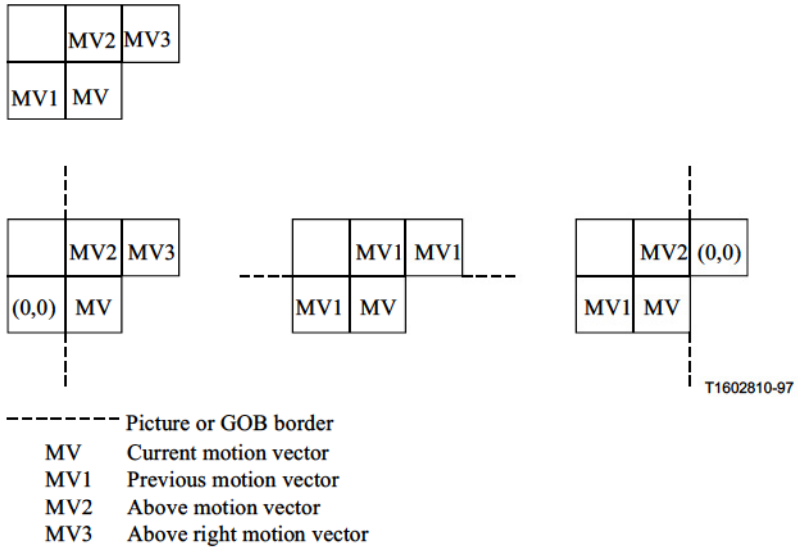



**Figure 12/H.263 – Motion vector prediction**

Advantage is taken of the fact that the range of motion vector component values is constrained. Each VLC word for MVD represents a pair of difference values. Only one of the pair will yield a macroblock vector component falling within the permitted range [−16, 15.5]. A positive value of the horizontal or vertical component of the motion vector signifies that the prediction is formed from pixels in the previous picture which are spatially to the right or below the pixels being predicted. If the unrestricted motion vector mode (see Annex D) is used, the decoding of motion vectors shall be performed as specified in D.2.

The motion vector is used for all pixels in all four luminance blocks in the macroblock. Motion vectors for both chrominance blocks are derived by dividing the component values of the macroblock vector by two, due to the lower chrominance format. The component values of the resulting quarter pixel resolution vectors are modified towards the nearest half pixel position as indicated in Table 18.

**Table 18/H.263 – Modification of quarter-pixel resolution chrominance vector components**

| **Quarter-pixel position** | 0 | 1/4 | 1/2 | 3/4 | 1 |
|---|---|---|---|---|---|
| **Resulting position** | 0 | 1/2 | 1/2 | 1/2 | 1 |

### 6.1.2 Interpolation for subpixel prediction

Half pixel values are found using bilinear interpolation as described in Figure 13. "/" indicates division by truncation.

The value of RCONTROL is equal to the value of the rounding type (RTYPE) bit (bit 6) in MPPTYPE (see 5.1.4.3), when the Source Format field (bits 6-8) in PTYPE indicates "extended PTYPE". Otherwise RCONTROL has an implied value of 0. Regardless of the RTYPE bit, the value of RCONTROL is set to 0 for the B-part of Improved PB frames (see Annex M).

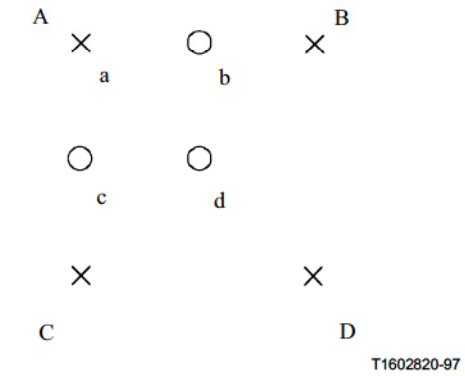



**Figure 13/H.263 – Half-pixel prediction by bilinear interpolation**

## 6.2 Coefficients decoding

### 6.2.1 Inverse quantization

The inverse quantization process is described in this subclause, except for when the optional Advanced INTRA Coding mode is in use (see Annex I). If LEVEL = "0", the reconstruction level REC = "0". The reconstruction level of INTRADC is given by Table 15. The reconstruction levels of all non-zero coefficients other than the INTRADC one are given by the following formulas:

$$|REC| = QUANT \cdot (2 \cdot |LEVEL| + 1) \qquad \text{if QUANT = "odd"}$$

$$|REC| = QUANT \cdot (2 \cdot |LEVEL| + 1) - 1 \qquad \text{if QUANT = "even"}$$

Note that this process disallows even valued numbers. This has been found to prevent accumulation of IDCT mismatch errors. After calculation of |REC|, the sign is added to obtain REC: REC = sign(LEVEL) · |REC|

Sign(LEVEL) is given by the last bit of the TCOEF code (see Table 16) or by Table 17.

### 6.2.2 Clipping of reconstruction levels

After inverse quantization, the reconstruction levels of all coefficients other than the INTRADC one are clipped to the range –2048 to 2047.

### 6.2.3 Zigzag positioning

The quantized transform coefficients are placed into an 8 × 8 block according to the sequence given in Figure 14, unless the optional Advanced INTRA Coding mode is in use (see Annex I). Coefficient 1 is the dc-coefficient.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 6 | 7 | 15 | 16 | 28 | 29 |
| 3 | 5 | 8 | 14 | 17 | 27 | 30 | 43 |
| 4 | 9 | 13 | 18 | 26 | 31 | 42 | 44 |
| 10 | 12 | 19 | 25 | 32 | 41 | 45 | 54 |
| 11 | 20 | 24 | 33 | 40 | 46 | 53 | 55 |
| 21 | 23 | 34 | 39 | 47 | 52 | 56 | 61 |
| 22 | 35 | 38 | 48 | 51 | 57 | 60 | 62 |
| 36 | 37 | 49 | 50 | 58 | 59 | 63 | 64 |

**Figure 14/H.263 – Zigzag positioning of quantized transform coefficients**

### 6.2.4 Inverse transform

After inverse quantization and zigzag of coefficients, the resulting 8 × 8 blocks are processed by a separable two-dimensional inverse discrete cosine transform of size 8 by 8. The output from the inverse transform ranges from –256 to +255 after clipping to be represented with 9 bits. The transfer function of the inverse transform is given by:

$$f(x,y)=\frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7}C(u)C(v)F(u,v)\cos\left[\pi(2x+1)\frac{u}{16}\right]\cos\left[\pi(2y+1)\frac{v}{16}\right]$$

with $u, v, x, y = 0, 1, 2, ..., 7$

where:

$x, y$ = spatial coordinates in the pixel domain,

$u, v$ = coordinates in the transform domain;

$C(u)$ = $1/\sqrt{2}$ for $u = 0$, otherwise 1;

$C(v)$ = $1/\sqrt{2}$ for $v = 0$, otherwise 1.

NOTE – Within the block being transformed, $x = 0$ and $y = 0$ refer to the pixel nearest the left and top edges of the picture respectively.

The arithmetic procedures for computing the inverse transform are not defined, but should meet the error tolerance specified in Annex A.

## 6.3 Reconstruction of blocks

### 6.3.1 Summation

After motion compensation and coefficients decoding (inverse transform included), a reconstruction is formed for each luminance and chrominance block. For INTRA blocks, the reconstruction is equal to the result of the inverse transformation. For INTER blocks, the reconstruction is formed by

summing the prediction and the result of the inverse transformation. The summation is performed on a pixel basis. For the summation in the Reduced-Resolution Update mode, refer to Annex Q.

### 6.3.2 Clipping

To prevent quantization distortion of transform coefficient amplitudes causing arithmetic overflow in the encoder and decoder loops, clipping functions are inserted. The clipper operates after the summation of prediction and reconstructed prediction error on resulting pixel values less than 0 or greater than 255, changing them to 0 and 255 respectively.

## ANNEX A

## Inverse transform accuracy specification

**A.1** Generate random integer pixel data values in the range –L to +H according to the random number generator given below ("C" version). Arrange into 8 by 8 blocks. Data set of 10 000 blocks should each be generated for (L = 256, H = 255), (L = H = 5) and (L = H = 300).

**A.2** For each 8 by 8 block, perform a separable, orthonormal, matrix multiply, forward discrete cosine transform using at least 64-bit floating point accuracy.

$$F(u,v)=\frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7}f(x,y)\cos\left[\pi(2x+1)\frac{u}{16}\right]\cos\left[\pi(2y+1)\frac{v}{16}\right]$$

with $u, v, x, y = 0, 1, 2, ..., 7$

where:

$x, y$ = spatial coordinates in the pixel domain;

$u, v$ = coordinates in the transform domain;

$C(u)$ = $1/\sqrt{2}$ for $u = 0$, otherwise 1;

$C(v)$ = $1/\sqrt{2}$ for $v = 0$, otherwise 1.

**A.3** For each block, round the 64 resulting transformed coefficients to the nearest integer values. Then clip them to the range –2048 to +2047. This is the 12-bit input data to the inverse transform.

**A.4** For each 8 by 8 block of 12-bit data produced by A.3, perform a separable, orthonormal, matrix multiply, inverse discrete cosine transform (IDCT) using at least 64-bit floating point accuracy. Round the resulting pixels to the nearest integer and clip to the range –256 to +255. These blocks of 8 × 8 pixels are the reference IDCT output data.

**A.5** For each 8 by 8 block produced by A.3, apply the IDCT under test and clip the output to the range –256 to +255. These blocks of 8 × 8 pixels are the test IDCT output data.

**A.6** For each of the 64 IDCT output pixels, and for each of the 10 000 block data sets generated above, measure the peak, mean and mean square error between the reference and the test data.

**A.7**

- For any pixel, the peak error should not exceed 1 in magnitude.
- For any pixel, the mean square error should not exceed 0.06.
- Overall, the mean square error should not exceed 0.02.
- For any pixel, the mean error should not exceed 0.015 in magnitude.
- Overall, the mean error should not exceed 0.0015 in magnitude.

**A.8** All zeros in shall produce all zeros out.

**A.9** Re-run the measurements using exactly the same data values of A.1, but change the sign on each pixel.

*"C" program for random number generation*

```
/* L and H shall be long, that is 32 bits */
long rand (L,H)
long      L,H;
{
     static long randx = 1;                    /* long is 32 bits            */
     static double z = (double) 0x7fffffff;
     long i,j;
     double    x;                              /* double is 64 bits          */
     randx = (randx * 1103515245) + 12345;
     i = randx & 0x7ffffffe;                   /* keep 30 bits               */
     x = ( (double)i ) /  z;                   /* range 0 to 0.99999 ...     */
     x *= (L+H+1);                             /* range 0 to < L+H+1         */
     j = x;                                    /* truncate to integer        */
     return(j - L);                            /* range -L to H              */
}
```

ANNEX B

## Hypothetical Reference Decoder

The Hypothetical Reference Decoder (HRD) is defined as follows:

**B.1** The HRD and the encoder have the same clock frequency as well as the same picture clock frequency, and are operated synchronously.

**B.2** The HRD receiving buffer size is (B + BPPmaxKb * 1024 bits) where (BPPmaxKb * 1024) is the maximum number of bits per picture that has been negotiated for use in the bitstream (see 3.6). The value of B is defined as follows:

$$B = 4 \cdot R_{max} / PCF$$

where PCF is the effective picture clock frequency, and $R_{max}$ is the maximum video bit rate during the connection in bits per second. The effective picture clock frequency is the standard CIF picture clock frequency unless a custom PCF is specified in the CPCFC field of the picture header. This value for B is a minimum. An encoder may use a larger value for B, provided the larger number is first negotiated by external means, for example Recommendation H.245.

The value for $R_{max}$ depends on the system configuration (for example GSTN or ISDN, single or multi-link) and may be equal to the maximum bit rate supported by the physical link. Negotiation of $R_{max}$ is done by external means (for example, Recommendation H.245).

**B.3** The HRD is initially empty.

**B.4** The HRD buffer is examined at picture clock intervals (1000/PCF ms). If at least one complete coded picture is in the buffer, then all the data for the earliest picture in bitstream order is instantaneously removed (e.g. at $t_{n+1}$ in Figure B.1). Immediately after removing the above data the buffer occupancy must be less than B. This is a requirement on the coder output bitstream including coded picture data and MCBPC and STUF stuffing but not error correction framing bits, fill indicator (Fi), fill bits or error correction parity information described in Annex H.

For the purposes of this definition, unless the optional Temporal, SNR, and Spatial Scalability mode is in use, a complete coded picture is one normal I- or P-picture, or a PB-frame or Improved PB-frame.

When the Temporal, SNR, and Spatial Scalability mode is in use (see Annex O), each enhancement layer is given an additional HRD, for which a complete coded picture is one EI-, EP-, or B-picture. The base layer buffer shall hold the bits of the picture header as they arrive, until a sufficient amount of the picture header has arrived to determine whether the picture is a base layer or enhancement layer picture, and the number of the enhancement layer. When it can be determined that the arriving picture belongs to an enhancement layer, all bits for that picture shall be instantly transferred to the appropriate enhancement layer HRD, and any later bits which arrive continue to be placed into the enhancement layer HRD until a sufficient amount of some new picture header has arrived to determine that the bitstream should again be re-routed into another HRD buffer. The process of enhancement layer identification is instantaneous and asynchronous, independent of the picture clock interval checking times.

To meet this requirement the number of bits for the $(n+1)$th coded picture $d_{n+1}$ must satisfy:

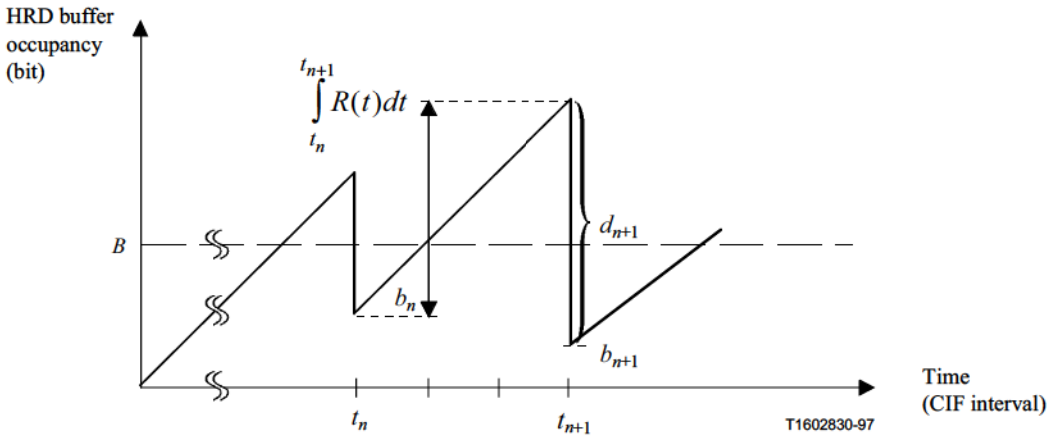
$$d_{n+1} \geq b_n + \int_{t_n}^{t_{n+1}} R(t)dt - B$$

where:

$b_n$ is the buffer occupancy just after time $t_n$;

$t_n$ is the time the $n$th coded picture is removed from the HRD buffer;

$R(t)$ is the video bit rate at time $t$.




NOTE – Time $(t_{n+1} - t_n)$ is an integer number of CIF picture periods (1/29.97, 2/29.97, 3/29.97,...).

**Figure B.1/H.263 – HRD buffer occupancy**

ANNEX C

## Considerations for multipoint

The following facilities are provided to support switched multipoint operation.

### C.1 Freeze picture request

Causes the decoder to freeze its displayed picture until a freeze picture release signal is received or a time-out period of at least six seconds has expired. The transmission of this signal is by external means (for example, Recommendation H.245). Note that a similar command may also be sent using supplemental enhancement information within the picture header of the video bitstream (see L.4).

### C.2 Fast update request

Causes the encoder to encode its next picture in INTRA mode with coding parameters such as to avoid buffer overflow. The transmission method for this signal is by external means (for example Recommendation H.245).

### C.3 Freeze picture release

A signal from an encoder which has responded to a fast update request and allows a decoder to exit from its freeze picture mode and display decoded pictures in the normal manner. This signal is transmitted by PTYPE (see 5.1.3) in the picture header of the first picture coded in response to the fast update request.

### C.4 Continuous Presence Multipoint and Video Multiplexing (CPM)

NOTE – Not used for Recommendation H.324.

In this Recommendation a negotiable Continuous Presence Multipoint and Video Multiplexing mode is provided in which up to four independent H.263 bitstreams can be multiplexed as independent "Sub-Bitstreams" in one new video bitstream with use of the PSBI, GSBI, SSBI, and ESBI fields. Capability exchange for this mode is done by external means (for example Recommendation H.242). Reference Picture Selection back-channel data for responding to independent Sub-Bitstreams is supported using the BCPM and BSBI fields.

When in CPM mode, the CPM field shall be set to "1" in each of the independent H.263 Sub-Bitstreams. Sub-Bitstreams are identified by a stream identifier number using the Sub-Bitstream Indicators (SBIs) in the Picture and GOB or slice and EOSBS headers of each H.263 bitstream. The SBI indicates the number of the H.263 bitstream to which that header and all following information until the next Picture or GOB header or slice header in the composed video bitstream belongs.

Each Sub-Bitstream is considered as a normal H.263 bitstream and shall therefore comply with the capabilities that are exchanged by external means. The information for the different H.263 bitstreams is not transmitted in any special predefined order, an SBI can have any value independent from preceding SBIs and the picture rates for the different H.263 bitstreams may be different. The information in each individual bitstream is also completely independent from the information in the other bitstreams. For example, the GFID codewords in one Sub-Bitstream are not influenced by GFID or PTYPE codewords in other Sub-Bitstreams. Similarly, the mode state inference rules when using an extended picture type (PLUSPTYPE) in the picture header and all other aspects of the video bitstream operation shall operate independently and separately for each Sub-Bitstream.

As a matter of convention, the Sub-Bitstream with the lowest Sub-Bitstream identifier number (sent in SBI) is considered to have the highest priority in situations in which some conflict of resource

requirements may necessitate a choice of priority (unless a different priority convention is established by external means).

In order to mark the end of each Sub-Bitstream of the CPM mode, a syntax is provided as shown in Figure C.1, provided the ability to send this added syntax is first negotiated by external means (while CPM operation was defined in Version 1 of this Recommendation, the end of Sub-Bitstream syntax was added in Version 2 and thus is not considered a part of Version 1 CPM operation). The end of Sub-Bitstream syntax (ESTUF + EOSBS + ESBI) marks the end of each Sub-Bitstream, rather than the end of the entire stream as done by an EOS code.

NOTE – No ability to negotiate CPM Sub-Bitstream operation as defined herein for Recommendation H.263 was adopted into any ITU-T Recommendations for terminals (such as Recommendation H.324) prior to the creation of Version 2 of this Recommendation. Thus any external negotiation of CPM operation adopted into a future H-Series terminal Recommendation shall imply support for the end of Sub-Bitstream syntax unless otherwise specified in the H-Series terminal Recommendation.

There are three parts to the end of Sub-Bitstream syntax. Following mandatory byte alignment using ESTUF, an EOSBS codeword of 23 bits is sent (corresponding to a GOB header with GN = 30, which is otherwise unused in the syntax, followed by a single zero-valued bit which is reserved for future use). The EOSBS codeword is then followed by a two-bit ESBI codeword indicating which Sub-Bitstream is affected. This pair of codewords signifies that the sending of data for the associated Sub-Bitstream has ended and that any subsequent data sent for the same Sub-Bitstream shall be entirely independent from that which came before the EOSBS. In particular, the next picture for the Sub-Bitstream after the EOSBS code shall not be an INTER picture or any other picture type which can use forward temporal prediction (an I- or EI-picture is allowed, but a P-picture, PB-frame, Improved PB-frame, B-picture, or EP-picture is not).

The syntax of EOSBS and ESBI is described in the following subclauses. ESTUF is described in 5.1.26.

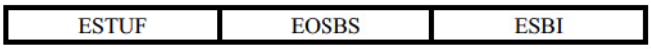



**Figure C.1/H.263 – Syntax diagram for End of Sub-Bitstream Indicators**

### C.4.1 End Of Sub-Bitstream code (EOSBS) (23 bits)

The EOSBS code is a codeword of 23 bits. Its value is 0000 0000 0000 0000 1 11110 0. It is up to the encoder whether to insert this codeword or not. EOSBS should not be sent unless at least one picture header has previously been sent for the same Sub-Bitstream indicated in the following ESBI field, and shall not be sent unless the capability to send EOSBS has been negotiated by external means. EOSBS codes shall be byte aligned. This is achieved by inserting ESTUF before the EOSBS start code such that the first bit of the EOSBS start code is the first (most significant) bit of a byte (see 5.1.26).

The EOSBS code indicates that the sending of data for the indicated Sub-Bitstream has stopped and the Sub-Bitstream has been declared over, until re-started by issuance of another picture start code for that Sub-Bitstream. Subsequent pictures with the same Sub-Bitstream identifier number (ESBI) shall be completely independent of and shall not depend in any way on the pictures which were sent prior to the EOSBS code.

Control and other information associated with the video bitstream in general without specification of which Sub-Bitstreams these codes are meant to apply (such as a Freeze Picture Request or a Fast Update Request sent in H.242) should be presumed to apply to all of the *active* Sub-Bitstreams only. A Sub-Bitstream is considered active if at least one picture start code has been received for the

Sub-Bitstream and the last data sent which was applicable to that Sub-Bitstream was not an EOS or EOSBS + ESBI.

### C.4.2 Ending Sub-Bitstream Indicator (ESBI) (2 bits)

ESBI is a fixed length codeword of two bits which follows immediately after EOSBS. It indicates the Sub-Bitstream number of the ending Sub-Bitstream. Its value is the natural two-bit binary representation of the Sub-Bitstream number.

## ANNEX D

## Unrestricted Motion Vector mode

This annex describes the optional Unrestricted Motion Vector mode of this Recommendation. The capability of this H.263 mode is signalled by external means (for example, Recommendation H.245). The use of this mode is indicated in PTYPE or PLUSPTYPE.

The range of motion vectors and the VLC table used for coding the motion vector differences for the Unrestricted Motion Vector mode depend on whether the PLUSPTYPE field of the picture header is present. When PLUSPTYPE is present, the range of motion vectors also depends on the picture size and the value of the UUI field in the picture header.

### D.1 Motion vectors over picture boundaries

In the default prediction mode of this Recommendation, motion vectors are restricted such that all pixels referenced by them are within the coded picture area (see 4.2.3). In the Unrestricted Motion Vector mode however this restriction is removed and therefore motion vectors *are* allowed to point outside the picture. When a pixel referenced by a motion vector is outside the coded picture area, an edge pixel is used instead. This edge pixel is found by limiting the motion vector to the last full-pixel position inside the coded picture area. Limitation of the motion vector is performed on a pixel basis and separately for each component of the motion vector.

For example, if the Unrestricted Motion Vector mode is used for a QCIF picture, the referenced pixel value for the luminance component is given by the following formula:

$$\mathrm{Rumv}(x, y) = R(x', y')$$

where:

$x, y, x', y'$ = spatial coordinates in the pixel domain;

$\mathrm{Rumv}(x, y)$ = pixel value of the referenced picture at (x, y) when in Unrestricted Motion Vector mode;

$R(x', y')$ = pixel value of the referenced picture at (x′, y′) when in Unrestricted Motion Vector mode;

$$x' \begin{cases} = 0 & \text{if } x < 0; \\ = 175 & \text{if } x > 175; \\ = x & \text{otherwise;} \end{cases}$$

$$y' \begin{cases} = 0 & \text{if } y < 0; \\ = 143 & \text{if } y > 143; \\ = y & \text{otherwise;} \end{cases}$$

and the coded picture area of R(x′, y′) is $0 \le x' \le 175$, $0 \le y' \le 143$. The given boundaries are integer pixel positions; however, (x′, y′) can also be a half-pixel position within these boundaries.

### D.1.1 Restrictions for motion vector values

If PLUSPTYPE is present in the picture header, the motion vector values are restricted such that no element of the 16 × 16 or (8 × 8) region that is selected shall have a horizontal or vertical distance more than 15 pixels outside the coded picture area. Note that this is a smaller extrapolation range than when PLUSPTYPE is not present.

NOTE 1 – When PLUSPTYPE is absent, the extrapolation range is a maximum of 31.5 pixels outside the coded picture area when the Unrestricted Motion Vector mode is used, and 16 pixels outside the coded picture area when the Advanced Prediction mode (see Annex F) is used without the Unrestricted Motion Vector mode.

NOTE 2 – When the Advanced Prediction mode (see Annex F) is in use, the motion vector for each 16 × 16 or (8 × 8) region affects a larger areas, due to overlapped block motion compensation. This can cause the effective extrapolation range to increase for the "remote" motion vectors of the Advanced Prediction mode, since the amount of overlapping (4 pixels, or 8 pixels when the Reduced-Resolution Update mode is also in use) adds to the amount of extrapolation required (even though the range of values allowed for each motion vector remains the same as when the Advanced Prediction mode is not in use).

## D.2 Extension of the motion vector range

In the default prediction mode, the values for both horizontal and vertical components of the motion vectors are restricted to the range [−16, 15.5] (this is also valid for the forward and backward motion vector components for B-pictures). In the Unrestricted Motion Vector mode, however, the maximum range for vector components is extended.

If the PLUSPTYPE field is not present in the picture header, the motion vector range is extended to [−31.5, 31.5], with the restriction that only values that are within a range of [−16, 15.5] around the predictor for each motion vector component can be reached if the predictor is in the range [−15.5, 16]. If the predictor is outside [−15.5, 16], all values within the range [−31.5, 31.5] with the same sign as the predictor plus the zero value can be reached. So, if $MV_c$ is the motion vector component and $P_c$ is the predictor for it, then:

$$-31.5 \le MV_c \le 0 \qquad if -31.5 \le P_c \le -16$$

$$-16 + P_c \le MV_c \le 15.5 + P_c \qquad if -15.5 \le P_c \le 16$$

$$0 \le MV_c \le 31.5 \qquad if\ \ 16.5 \le P_c \le 31.5$$

In the Unrestricted Motion Vector mode, the interpretation of Table 14 for MVD, $MVD_{2\text{-}4}$ and MVDB is as follows:

- If the predictor for the motion vector component is in the range [−15.5, 16], only the first column of vector differences applies.
- If the predictor for the motion vector component is outside the range [−15.5, 16], the vector difference from Table 14 shall be used that results in a vector component inside the range [−31.5, 31.5] with the same sign as the predictor (including zero).

The predictor for MVD and $MVD_{2\text{-}4}$ is defined as the median value of the vector components MV1, MV2 and MV3 as defined in 6.1.1 and F.2. For MVDB, the predictor $P_c = (TR_B \times MV)/TR_D$, where MV represents a vector component for an 8 * 8 luminance block in a P-picture (see also G.4).

If PLUSPTYPE is present, the motion vector range does not depend on the motion vector prediction value. If the UUI field is set to "1" the motion vector range depends on the picture format. For standardized picture formats up to CIF the range is [−32, 31.5], for those up to 4CIF the range is

[−64, 63.5], and for those up to 16CIF the range is [−128, 127.5], and for even larger custom picture formats the range is [−256, 255.5]. The horizontal and vertical motion vector ranges may be different for custom picture formats. The horizontal and vertical ranges are specified in Tables D.1 and D.2.

**Table D.1/H.263 – Horizontal motion vector range when PLUSPTYPE present and UUI = 1**

| Picture width | Horizontal motion vector range |
|---|---|
| 4, …, 352 | [–32, 31.5] |
| 356, …, 704 | [–64, 63.5] |
| 708, …, 1408 | [–128, 127.5] |
| 1412, …, 2048 | [–256, 255.5] |

**Table D.2/H.263 – Vertical motion vector range when PLUSPTYPE present and UUI = 1**

| Picture height | Vertical motion vector range |
|---|---|
| 4, …, 288 | [–32, 31.5] |
| 292, …, 576 | [–64, 63.5] |
| 580, …, 1152 | [–128, 127.5] |

In the Reduced-Resolution Update mode, the specified range applies to the pseudo motion vectors. This implies that the resulting actual motion vector range is enlarged to approximately double size (see also Annex Q).

If UUI is set to "01", the motion vectors are not limited except by their distance to the coded area border as explained in D.1.1. The same limitation applies to the actual motion vectors (not just the pseudo motion vectors) in the Reduced-Resolution Update mode.

To encode the motion vectors when PLUSPTYPE is present, Table D.3 is used to encode the difference between the motion vector and the motion vector prediction. Every entry in Table D.3 has a single value (in contrast to Table 14). The motion vector range and the use of Table D.3 to encode motion vector data apply to all picture types when PLUSPTYPE is present.

Motion vectors differences are always coded as a pair of a horizontal and a vertical component. If a pair equals (0.5, 0.5) six consecutive zeros are produced. To prevent start code emulation, this occurrence shall be followed by one bit set to "1". This corresponds to sending one additional zero motion vector component.

**Table D.3/H.263 – Motion Vector Table used when PLUSPTYPE present**

| Absolute value of vector difference in half-pixel units | Number of bits | Codes |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 3 | 0s0 |
| "$x_0$"+2 (2:3) | 5 | 0$x_0$1s0 |
| "$x_1x_0$"+4 (4:7) | 7 | 0$x_1$1$x_0$1s0 |
| "$x_2x_1x_0$"+8 (8:15) | 9 | 0$x_2$1$x_1$1$x_0$1s0 |
| "$x_3x_2x_1x_0$"+16 (16:31) | 11 | 0$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_4x_3x_2x_1x_0$"+32 (32:63) | 13 | 0$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_5x_4x_3x_2x_1x_0$"+64 (64:127) | 15 | 0$x_5$1$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_6x_5x_4x_3x_2x_1x_0$"+128 (128:255) | 17 | 0$x_6$1$x_5$1$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_7x_6x_5x_4x_3x_2x_1x_0$"+256 (256:511) | 19 | 0$x_7$1$x_6$1$x_5$1$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_8x_7x_6x_5x_4x_3x_2x_1x_0$"+512 (512:1023) | 21 | 0$x_8$1$x_7$1$x_6$1$x_5$1$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_9x_8x_7x_6x_5x_4x_3x_2x_1x_0$"+1024 (1024:2047) | 23 | 0$x_9$1$x_8$1$x_7$1$x_6$1$x_5$1$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |
| "$x_{10}x_9x_8x_7x_6x_5x_4x_3x_2x_1x_0$"+2048 (2048:4095) | 25 | 0$x_{10}$1$x_9$1$x_8$1$x_7$1$x_6$1$x_5$1$x_4$1$x_3$1$x_2$1$x_1$1$x_0$1s0 |

Table D.3 is a regularly constructed reversible table. Each row represents an interval of motion vector differences in half-pixel units. The bits "…$x_1x_0$" denote all bits following the leading "1" in the binary representation of the absolute value of the motion vector difference. The bit "s" denotes the sign of the motion vector difference, "0" for positive and "1" for negative. The binary representation of the motion vector difference is interleaved with bits that indicate if the code continues or ends. For example, the motion vector difference −13 has sign s = 1 and binary representation 1$x_2x_1x_0$ = 1101. It is encoded as 0 $x_2$1 $x_1$1 $x_0$1 s0 = 0 11 01 11 10. The 0 in the second position of the last group of two bits indicates the end of the code.

# ANNEX E

## Syntax-based Arithmetic Coding mode

### E.1 Introduction

In the Variable Length Coding/Decoding (VLC/VLD) as described in clause 5, a symbol is VLC encoded using a specific table based on the syntax of the coder. This table typically stores lengths and values of the VLC code words. The symbol is mapped to an entry of the table in a table look-up operation, and then the binary code word specified by the entry is sent out normally to a buffer for transmitting to the receiver. In VLD decoding, the received bitstream is matched entry by entry in a specific table based on the syntax of the coder. This table must be the same as the one used in the encoder for encoding the current symbol. The matched entry in the table is then mapped back to the corresponding symbol which is the end result of the VLD decoder and is then used for recovering the video pictures. This VLC/VLD process implies that each symbol must be encoded into a fixed integral number of bits. Removing this restriction of fixed integral number of bits for symbols can lead to reductions of resulting bit rates, which can be achieved by arithmetic coding.

This annex describes the optional Syntax-based Arithmetic Coding (SAC) mode of this Recommendation. In this mode, all the corresponding variable length coding/decoding operations of this Recommendation are replaced with arithmetic coding/decoding operations. The capability of this H.263 mode is signalled by external means (for example, Recommendation H.245). The use of this mode is indicated in PTYPE.

## E.2 Specification of SAC encoder

In SAC mode, a symbol is encoded by using a specific array of integers (or a model) based on the syntax of the coder and by calling the following procedure which is specified in C language.

```
#define  q1   16384
#define  q2   32768
#define  q3   49152
#define  top  65535

static long  low, high, opposite_bits, length;
void     encode_a_symbol(int index, int cumul_freq[ ])
{
       length = high - low + 1;
       high = low - 1 + (length * cumul_freq[index]) / cumul_freq[0];
       low += (length * cumul_freq[index+1]) / cumul_freq[0];
       for ( ; ; ) {
          if (high < q2) {
             send out a bit "0" to PSC_FIFO;
             while (opposite_bits > 0) {
                send out a bit "1" to PSC_FIFO;
                opposite_bits--;
             }
          }
          else if (low >= q2) {
             send out a bit "1" to PSC_FIFO;
             while (opposite_bits > 0) {
                send out a bit "0" to PSC_FIFO;
                opposite_bits--;
             }
             low -= q2;
             high -= q2;
          }
          else if (low >= q1 && high < q3) {
             opposite_bits += 1;
             low -= q1;
             high -= q1;
          }
          else break;

          low *= 2;
          high = 2 * high+1;
       }
}
```

The values of `low`, `high` and `opposite_bits` are initialized to `0`, `top` and `0`, respectively. `PSC_FIFO` is a FIFO for buffering the output bits from the arithmetic encoder. The model is specified through `cumul_freq[ ]`, and the symbol is specified using its index in the model.

## E.3 Specification of SAC decoder

In SAC decoder, a symbol is decoded by using a specific model based on the syntax and by calling the following procedure which is specified in C language.

```
static long    low, high, code_value, bit, length, index, cum;
int   decode_a_symbol(int cumul_freq[ ])
{
      length = high - low + 1;
      cum = (-1 + (code_value - low + 1) * cumul_freq[0]) / length;
      for (index = 1; cumul_freq[index] > cum; index++);
      high = low - 1 + (length * cumul_freq[index-1]) / cumul_freq[0];
      low += (length * cumul_freq[index]) / cumul_freq[0];
      for ( ; ; ) {
         if (high < q2);
         else if (low >= q2) {
            code_value -= q2;
            low -= q2;
            high -= q2;
         }
         else if (low >= q1 && high < q3) {
            code_value -= q1;
            low -= q1;
            high -= q1;
         }
         else break;

         low *= 2;
         high = 2 * high + 1;
         get bit from PSC_FIFO;
         code_value = 2 * code_value + bit;
      }
      return (index-1);
}
```

Again the model is specified through `cumul_freq[ ]`. The decoded symbol is returned through its index in the model. `PSC_FIFO` is a FIFO for buffering the incoming bitstream. The decoder is initialized to start decoding an arithmetic coded bitstream by calling the following procedure.

```
void  decoder_reset( )
{
      code_value = 0;
      low = 0;
      high = top;
      for (int i = 1;   i <= 16;   i++) {
         get bit from PSC_FIFO;
         code_value = 2 * code_value + bit;
      }
}
```

## E.4 Syntax

As in VLC table mode of this Recommendation, the syntax of the symbols is partitioned into four layers: Picture, Group of Blocks, Macroblock and Block. The syntax of the top three layers remain exactly the same. The syntax of the Block layer also remains quite similar, but is illustrated in Figure E.1.

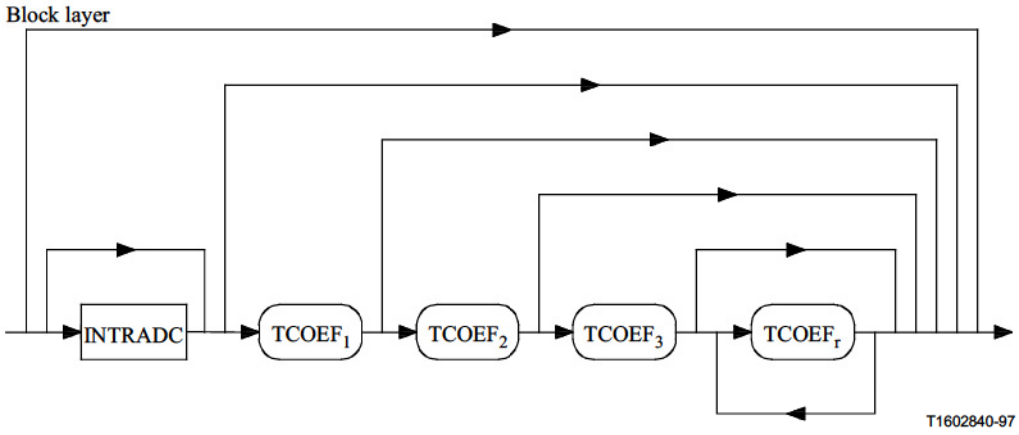



**Figure E.1/H.263 – Structure of SAC Block layer**

In Figure E.1, TCOEF1, TCOEF2, TCOEF3 and TCOEFr are Last-Run-Level symbols as defined in 5.4.2, and are the possible 1st, 2nd, 3rd and rest of the symbols, respectively. TCOEF1, TCOEF2, TCOEF3 and TCOEFr are only present when one, two, three or more coefficients are present in the block layer, respectively.

## E.5 PSC_FIFO

PSC_FIFO in encoder or decoder is a FIFO of size > 17 bits. In PSC_FIFO of encoder, illegal emulations of PSC and GBSC are located and are avoided by stuffing a "1" after each successive appearance of 14 "0"s (which are not part of PSC or GBSC). In PSC_FIFO of the decoder, the first "1" after each string of 14 "0"s is deleted; if instead a string of 14 "0"s is followed by a "0", it indicates that a legal PSC or GBSC is detected. The exact location of the PSC or GBSC is determined by the next "1" following the string of zeros.

## E.6 Header layer symbols

The header layers of the syntax are considered to be those syntactical elements above the block and macroblock layers (see Figure 6 and the syntax specification in the text). The header levels of the basic Version 1 syntax can form three possible strings, (PSTUF)--PSC--TR--PTYPE--PQUANT--CPM--(PSBI)--(TRB-DBQUANT)--PEI--(PSUPP--PEI--...), (GSTUF)--GBSC--GN--(GSBI)--GFID--GQUANT, and (ESTUF)--(EOS)--(PSTUF). In the revised syntax of Version 2, the header levels of the syntax can have other structures (see Figure 6 and the syntax specification in the text). The header level syntax strings are directly sent to PSC_FIFO as in the normal VLC table mode of this Recommendation at encoder side, and are directly sent out from PSC_FIFO in the decoder after a legal PSC, GBSC, SSC, EOS, or EOSBS is detected.

If a header is not the first in a video session, the arithmetic encoder needs to be reset before sending the header by calling the following procedure. This procedure shall also be called at the end of a video session if (ESTUF)--EOS [or (ESTUF)--EOSBS for the Sub-Bitstream for which the last header was sent] is not sent.

```
void encoder_flush( )
{
      opposite_bits++;
      if (low < q1) {
         send out a bit "0" to PSC_FIFO;
         while (opposite_bits > 0) {
            send out a bit "1" to PSC_FIFO;
            opposite_bits--;
         }
      }
      else {
         send out a bit "1" to PSC_FIFO;
         while (opposite_bits > 0) {
            send out a bit "0" to PSC_FIFO;
            opposite_bits--;
         }
      }
      low = 0;
      high = top;
}
```

In the decoder, after each fixed length symbol string, procedure **`decoder_reset`** is called.

### E.7 Macroblock and Block layer symbols

Models for the macroblock and block layer symbols are included in E.8. The indices as given in the VLC tables of clause 5 are used for the indexing of the integers in the models.

The model for COD in P-pictures is named cumf_COD. The index for COD being "0" is 0, and is 1 for COD being "1". The model for MCBPC in P-pictures is named cumf_MCBPC_no4MVQ unless PLUSPTYPE is present in the picture header and either the Advanced Prediction mode (Annex F) or the Deblocking Filter mode (Annex J) is in use, in which case the model is named cumf_MCBPC_4MVQ The indexes for MCBPC are defined in Table 7 for I-pictures and Table 8 for P-pictures. The model for MCBPC in I-pictures is named by cumf_MCBPC_intra.

The model for MODB is cumf_MODB_G if Annex G is used or cumf_MODB_M if Annex M is used. The indexes for MODB are defined in Table 11 or Table M.1, respectively. The model for CBPBn, n = 1, 2, ..., 4, is cumf_YCBPB, and the model for CBPBn, n = 5, 6, is cumf_UVCBPB, with index 0 for CBPBn = 0 and index 1 for CBPBn = 1.

The model for CBPY is cumf_CBPY in INTER macroblocks and cumf_CBPY_intra in INTRA macroblocks. The model for DQUANT is cumf_DQUANT. The indexing for CBPY and DQUANT is defined in Tables 13 and 12, respectively.

The model for MVD, $MVD_{2\text{-}4}$ and MVDB is cumf_MVD and the model for INTRADC is cumf_INTRADC. The indexing is defined in Tables 14 and 15, respectively.

A non-escaped TCOEF consists of a symbol for TCOEF1/2/3/r followed by a symbol, SIGN, for the sign of the TCOEF. The models for TCOEF1, TCOEF2, TCOEF3 and TCOEFr in INTER blocks are cumf_TCOEF1, cumf_TCOEF2, cumf_TCOEF3, cumf_TCOEFr. The models for INTRA blocks are cumf_TCOEF1_intra, cumf_TCOEF2_intra, cumf_TCOEF3_intra, cumf_TCOEFr_intra. For all TCOEFs the indexing is defined in Table 16. The model for SIGN is cumf_SIGN. The indexing for SIGN is 0 for positive sign and 1 for negative sign.

The models for LAST, RUN, LEVEL after ESCAPE are cumf_LAST (cumf_LAST_intra), cumf_RUN (cumf_RUN_intra), cumf_LEVEL (cumf_LEVEL_intra) for INTER (INTRA) blocks. The indexing for LAST is 0 for LAST = 0 and 1 for LAST = 1, while the indexing for RUN and LEVEL is defined in Table 17.

The model for INTRA_MODE is cumf_INTRA_AC_DC. The indexing is defined in Table I.1.

### E.8 SAC models

int cumf_COD[3]={16383, 6849, 0};

int cumf_MCBPC_no4MVQ[22]={16383, 4105, 3088, 2367, 1988, 1621, 1612, 1609, 1608, 496, 353, 195, 77, 22, 17, 12, 5, 4, 3, 2, 1, 0};

int cumf_MCBPC_4MVQ[26]={16383, 6880, 6092, 5178, 4916, 3965, 3880, 3795, 3768, 1491, 1190, 889, 655, 442, 416, 390, 360, 337, 334, 331, 327, 326, 88, 57, 26, 0};

int cumf_MCBPC_intra[10]={16383, 7410, 6549, 5188, 442, 182, 181, 141, 1, 0};

int cumf_MODB_G[4]={16383, 6062, 2130, 0};

int cumf_MODB_M[7] = {16383, 6717, 4568, 2784, 1370, 655, 0};

int cumf_YCBPB[3]={16383, 6062, 0};

int cumf_UVCBPB[3]={16383, 491, 0};

int cumf_CBPY[17]={16383, 14481, 13869, 13196, 12568, 11931, 11185, 10814, 9796, 9150, 8781, 7933, 6860, 6116, 4873, 3538, 0};

int cumf_CBPY_intra[17]={16383, 13619, 13211, 12933, 12562, 12395, 11913, 11783, 11004, 10782, 10689, 9928, 9353, 8945, 8407, 7795, 0};

int cumf_DQUANT[5]={16383, 12287, 8192, 4095, 0};

int cumf_MVD[65]={16383, 16380, 16369, 16365, 16361, 16357, 16350, 16343, 16339, 16333, 16326, 16318, 16311, 16306, 16298, 16291, 16283, 16272, 16261, 16249, 16235, 16222, 16207, 16175, 16141, 16094, 16044, 15936, 15764, 15463, 14956, 13924, 11491, 4621, 2264, 1315, 854, 583, 420, 326, 273, 229, 196, 166, 148, 137, 123, 114, 101, 91, 82, 76, 66, 59, 53, 46, 36, 30, 26, 24, 18, 14, 10, 5, 0};

int cumf_INTRADC[255]={16383, 16380, 16379, 16378, 16377, 16376, 16370, 16361, 16360, 16359, 16358, 16357, 16356, 16355, 16343, 16238, 16237, 16236, 16230, 16221, 16220, 16205, 16190, 16169, 16151, 16130, 16109, 16094, 16070, 16037, 16007, 15962, 15938, 15899, 15854, 15815, 15788, 15743, 15689, 15656, 15617, 15560, 15473, 15404, 15296, 15178, 15106, 14992, 14868, 14738, 14593, 14438, 14283, 14169, 14064, 14004, 13914, 13824, 13752, 13671, 13590, 13515, 13458, 13380, 13305, 13230, 13143, 13025, 12935, 12878, 12794, 12743, 12656, 12596, 12521, 12443, 12359, 12278, 12200, 12131, 12047, 12002, 11948, 11891, 11828, 11744, 11663, 11588, 11495, 11402, 11288, 11204, 11126, 11039, 10961, 10883, 10787, 10679, 10583, 10481, 10360, 10227, 10113, 9961, 9828, 9717, 9584, 9485, 9324, 9112, 9019, 8908, 8766, 8584, 8426, 8211, 7920, 7663, 7406, 7152, 6904, 6677, 6453, 6265, 6101, 5904, 5716, 5489, 5307, 5056, 4850, 4569, 4284, 3966, 3712, 3518, 3342, 3206, 3048, 2909, 2773, 2668, 2596, 2512, 2370, 2295, 2232, 2166, 2103, 2022, 1956, 1887, 1830, 1803, 1770, 1728, 1674, 1635, 1599, 1557, 1500, 1482, 1434, 1389, 1356, 1317, 1284, 1245, 1200, 1179, 1140, 1110, 1092, 1062, 1044, 1035, 1014, 1008, 993, 981, 954, 936, 912, 894, 876, 864, 849, 828, 816, 801, 792, 777, 756, 732, 690, 660, 642, 615, 597, 576, 555, 522, 489, 459, 435, 411, 405, 396, 387, 375, 360, 354, 345, 344, 329, 314, 293, 278, 251, 236, 230, 224, 215, 214, 208, 199, 193, 184, 178, 169, 154, 127, 100, 94, 73, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 20, 19, 18, 17, 16, 15, 9, 0};

int cumf_TCOEF1[104]={16383, 13455, 12458, 12079, 11885, 11800, 11738, 11700, 11681, 11661, 11651, 11645, 11641, 10572, 10403, 10361, 10346, 10339, 10335, 9554, 9445, 9427, 9419, 9006, 8968, 8964, 8643, 8627, 8624, 8369, 8354, 8352, 8200, 8192, 8191, 8039, 8036, 7920, 7917, 7800, 7793, 7730, 7727, 7674, 7613, 7564, 7513, 7484, 7466, 7439, 7411, 7389, 7373, 7369, 7359, 7348,

7321, 7302, 7294, 5013, 4819, 4789, 4096, 4073, 3373, 3064, 2674, 2357, 2177, 1975, 1798, 1618, 1517, 1421, 1303, 1194, 1087, 1027, 960, 890, 819, 758, 707, 680, 656, 613, 566, 534, 505, 475, 465, 449, 430, 395, 358, 335, 324, 303, 295, 286, 272, 233, 215, 0};

int cumf_TCOEF2[104]={16383, 13582, 12709, 12402, 12262, 12188, 12150, 12131, 12125, 12117, 12113, 12108, 12104, 10567, 10180, 10070, 10019, 9998, 9987, 9158, 9037, 9010, 9005, 8404, 8323, 8312, 7813, 7743, 7726, 7394, 7366, 7364, 7076, 7062, 7060, 6810, 6797, 6614, 6602, 6459, 6454, 6304, 6303, 6200, 6121, 6059, 6012, 5973, 5928, 5893, 5871, 5847, 5823, 5809, 5796, 5781, 5771, 5763, 5752, 4754, 4654, 4631, 3934, 3873, 3477, 3095, 2758, 2502, 2257, 2054, 1869, 1715, 1599, 1431, 1305, 1174, 1059, 983, 901, 839, 777, 733, 683, 658, 606, 565, 526, 488, 456, 434, 408, 380, 361, 327, 310, 296, 267, 259, 249, 239, 230, 221, 214, 0};

int cumf_TCOEF3[104]={16383, 13532, 12677, 12342, 12195, 12112, 12059, 12034, 12020, 12008, 12003, 12002, 12001, 10586, 10297, 10224, 10202, 10195, 10191, 9223, 9046, 8999, 8987, 8275, 8148, 8113, 7552, 7483, 7468, 7066, 7003, 6989, 6671, 6642, 6631, 6359, 6327, 6114, 6103, 5929, 5918, 5792, 5785, 5672, 5580, 5507, 5461, 5414, 5382, 5354, 5330, 5312, 5288, 5273, 5261, 5247, 5235, 5227, 5219, 4357, 4277, 4272, 3847, 3819, 3455, 3119, 2829, 2550, 2313, 2104, 1881, 1711, 1565, 1366, 1219, 1068, 932, 866, 799, 750, 701, 662, 605, 559, 513, 471, 432, 403, 365, 336, 312, 290, 276, 266, 254, 240, 228, 223, 216, 206, 199, 192, 189, 0};

int cumf_TCOEFr[104]={16383, 13216, 12233, 11931, 11822, 11776, 11758, 11748, 11743, 11742, 11741, 11740, 11739, 10203, 9822, 9725, 9691, 9677, 9674, 8759, 8609, 8576, 8566, 7901, 7787, 7770, 7257, 7185, 7168, 6716, 6653, 6639, 6276, 6229, 6220, 5888, 5845, 5600, 5567, 5348, 5327, 5160, 5142, 5004, 4900, 4798, 4743, 4708, 4685, 4658, 4641, 4622, 4610, 4598, 4589, 4582, 4578, 4570, 4566, 3824, 3757, 3748, 3360, 3338, 3068, 2835, 2592, 2359, 2179, 1984, 1804, 1614, 1445, 1234, 1068, 870, 739, 668, 616, 566, 532, 489, 453, 426, 385, 357, 335, 316, 297, 283, 274, 266, 259, 251, 241, 233, 226, 222, 217, 214, 211, 209, 208, 0};

int cumf_TCOEF1_intra[104]={16383, 13383, 11498, 10201, 9207, 8528, 8099, 7768, 7546, 7368, 7167, 6994, 6869, 6005, 5474, 5220, 5084, 4964, 4862, 4672, 4591, 4570, 4543, 4397, 4337, 4326, 4272, 4240, 4239, 4212, 4196, 4185, 4158, 4157, 4156, 4140, 4139, 4138, 4137, 4136, 4125, 4124, 4123, 4112, 4111, 4110, 4109, 4108, 4107, 4106, 4105, 4104, 4103, 4102, 4101, 4100, 4099, 4098, 4097, 3043, 2897, 2843, 1974, 1790, 1677, 1552, 1416, 1379, 1331, 1288, 1251, 1250, 1249, 1248, 1247, 1236, 1225, 1224, 1223, 1212, 1201, 1200, 1199, 1198, 1197, 1196, 1195, 1194, 1193, 1192, 1191, 1190, 1189, 1188, 1187, 1186, 1185, 1184, 1183, 1182, 1181, 1180, 1179, 0};

int cumf_TCOEF2_intra[104]={16383, 13242, 11417, 10134, 9254, 8507, 8012, 7556, 7273, 7062, 6924, 6839, 6741, 6108, 5851, 5785, 5719, 5687, 5655, 5028, 4917, 4864, 4845, 4416, 4159, 4074, 3903, 3871, 3870, 3765, 3752, 3751, 3659, 3606, 3580, 3541, 3540, 3514, 3495, 3494, 3493, 3474, 3473, 3441, 3440, 3439, 3438, 3425, 3424, 3423, 3422, 3421, 3420, 3401, 3400, 3399, 3398, 3397, 3396, 2530, 2419, 2360, 2241, 2228, 2017, 1687, 1576, 1478, 1320, 1281, 1242, 1229, 1197, 1178, 1152, 1133, 1114, 1101, 1088, 1087, 1086, 1085, 1072, 1071, 1070, 1069, 1068, 1067, 1066, 1065, 1064, 1063, 1062, 1061, 1060, 1059, 1058, 1057, 1056, 1055, 1054, 1053, 1052, 0};

int cumf_TCOEF3_intra[104]={16383, 12741, 10950, 10071, 9493, 9008, 8685, 8516, 8385, 8239, 8209, 8179, 8141, 6628, 5980, 5634, 5503, 5396, 5327, 4857, 4642, 4550, 4481, 4235, 4166, 4151, 3967, 3922, 3907, 3676, 3500, 3324, 3247, 3246, 3245, 3183, 3168, 3084, 3069, 3031, 3030, 3029, 3014, 3013, 2990, 2975, 2974, 2973, 2958, 2943, 2928, 2927, 2926, 2925, 2924, 2923, 2922, 2921, 2920, 2397, 2298, 2283, 1891, 1799, 1591, 1445, 1338, 1145, 1068, 1006, 791, 768, 661, 631, 630, 615, 592, 577, 576, 561, 546, 523, 508, 493, 492, 491, 476, 475, 474, 473, 472, 471, 470, 469, 468, 453, 452, 451, 450, 449, 448, 447, 446, 0};

int cumf_TCOEFr_intra[104]={16383, 12514, 10776, 9969, 9579, 9306, 9168, 9082, 9032, 9000, 8981, 8962, 8952, 7630, 7212, 7053, 6992, 6961, 6940, 6195, 5988, 5948, 5923, 5370, 5244, 5210, 4854, 4762, 4740, 4384, 4300, 4288, 4020, 3968, 3964, 3752, 3668, 3511, 3483, 3354, 3322, 3205, 3183, 3108, 3046, 2999, 2981, 2974, 2968, 2961, 2955, 2949, 2943, 2942, 2939, 2935, 2934, 2933, 2929, 2270, 2178, 2162, 1959, 1946, 1780, 1651, 1524, 1400, 1289, 1133, 1037, 942, 849, 763, 711, 591, 521, 503, 496, 474, 461, 449, 442, 436, 426, 417, 407, 394, 387, 377, 373, 370, 367, 366, 365, 364, 363, 362, 358, 355, 352, 351, 350, 0};

int cumf_SIGN[3]={16383, 8416, 0};

int cumf_LAST[3]={16383, 9469, 0};

int cumf_LAST_intra[3]={16383, 2820, 0};

int cumf_RUN[65]={16383, 15310, 14702, 13022, 11883, 11234, 10612, 10192, 9516, 9016, 8623, 8366, 7595, 7068, 6730, 6487, 6379, 6285, 6177, 6150, 6083, 5989, 5949, 5922, 5895, 5828, 5774, 5773, 5394, 5164, 5016, 4569, 4366, 4136, 4015, 3867, 3773, 3692, 3611, 3476, 3341, 3301, 2787, 2503, 2219, 1989, 1515, 1095, 934, 799, 691, 583, 435, 300, 246, 206, 125, 124, 97, 57, 30, 3, 2, 1, 0};

int cumf_RUN_intra[65]={16383, 10884, 8242, 7124, 5173, 4745, 4246, 3984, 3034, 2749, 2607, 2298, 966, 681, 396, 349, 302, 255, 254, 253, 206, 159, 158, 157, 156, 155, 154, 153, 106, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0};

int cumf_LEVEL[255]={16383, 16382, 16381, 16380, 16379, 16378, 16377, 16376, 16375, 16374, 16373, 16372, 16371, 16370, 16369, 16368, 16367, 16366, 16365, 16364, 16363, 16362, 16361, 16360, 16359, 16358, 16357, 16356, 16355, 16354, 16353, 16352, 16351, 16350, 16349, 16348, 16347, 16346, 16345, 16344, 16343, 16342, 16341, 16340, 16339, 16338, 16337, 16336, 16335, 16334, 16333, 16332, 16331, 16330, 16329, 16328, 16327, 16326, 16325, 16324, 16323, 16322, 16321, 16320, 16319, 16318, 16317, 16316, 16315, 16314, 16313, 16312, 16311, 16310, 16309, 16308, 16307, 16306, 16305, 16304, 16303, 16302, 16301, 16300, 16299, 16298, 16297, 16296, 16295, 16294, 16293, 16292, 16291, 16290, 16289, 16288, 16287, 16286, 16285, 16284, 16283, 16282, 16281, 16280, 16279, 16278, 16277, 16250, 16223, 16222, 16195, 16154, 16153, 16071, 15989, 15880, 15879, 15878, 15824, 15756, 15674, 15606, 15538, 15184, 14572, 13960, 10718, 7994, 5379, 2123, 1537, 992, 693, 611, 516, 448, 421, 380, 353, 352, 284, 257, 230, 203, 162, 161, 160, 133, 132, 105, 104, 103, 102, 101, 100, 99, 98, 97, 96, 95, 94, 93, 92, 91, 90, 89, 88, 87, 86, 85, 84, 83, 82, 81, 80, 79, 78, 77, 76, 75, 74, 73, 72, 71, 70, 69, 68, 67, 66, 65, 64, 63, 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0};

int cumf_LEVEL_intra[255]={16383, 16379, 16378, 16377, 16376, 16375, 16374, 16373, 16372, 16371, 16370, 16369, 16368, 16367, 16366, 16365, 16364, 16363, 16362, 16361, 16360, 16359, 16358, 16357, 16356, 16355, 16354, 16353, 16352, 16351, 16350, 16349, 16348, 16347, 16346, 16345, 16344, 16343, 16342, 16341, 16340, 16339, 16338, 16337, 16336, 16335, 16334, 16333, 16332, 16331, 16330, 16329, 16328, 16327, 16326, 16325, 16324, 16323, 16322, 16321, 16320, 16319, 16318, 16317, 16316, 16315, 16314, 16313, 16312, 16311, 16268, 16267, 16224, 16223, 16180, 16179, 16136, 16135, 16134, 16133, 16132, 16131, 16130, 16129, 16128, 16127, 16126, 16061, 16018, 16017, 16016, 16015, 16014, 15971, 15970, 15969, 15968, 15925, 15837, 15794, 15751, 15750, 15749, 15661, 15618, 15508, 15376, 15288, 15045, 14913, 14781, 14384, 13965, 13502, 13083, 12509, 12289, 12135, 11892, 11738, 11429, 11010, 10812, 10371, 9664, 9113, 8117, 8116, 8028, 6855, 5883, 4710, 4401, 4203, 3740, 3453, 3343, 3189, 2946, 2881, 2661, 2352, 2132, 1867, 1558, 1382, 1250, 1162, 1097, 1032, 967, 835, 681, 549, 439, 351, 350, 307, 306, 305, 304,

303, 302, 301, 300, 299, 298, 255, 212, 211, 210, 167, 166, 165, 164, 163, 162, 161, 160, 159, 158, 115, 114, 113, 112, 111, 68, 67, 66, 65, 64, 63, 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20, 19, 18, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0};

int cumf_INTRA_AC_DC[4]={16383, 9229, 5461, 0};

ANNEX F

**Advanced Prediction mode**

## F.1 Introduction

This annex describes the optional Advanced Prediction mode of this Recommendation, including overlapped block motion compensation and the possibility of four motion vectors per macroblock. The capability of this mode is signalled by external means (for example, Recommendation H.245). The use of this mode is indicated in PTYPE. In the Advanced Prediction mode, motion vectors are allowed to cross picture boundaries as is the case in the Unrestricted Motion Vector mode (for the description of this technique, refer to D.1). The extended motion vector range feature of the Unrestricted Motion Vector mode is not automatically included in the Advanced Prediction mode, and only is active if the Unrestricted Motion Vector mode is selected. If the Advanced Prediction mode is used in combination with the PB-frames mode, overlapped motion compensation is only used for prediction of the P-pictures, not for the B-pictures.

## F.2 Four motion vectors per macroblock

In this Recommendation, one motion vector per macroblock is used except when in Advanced Prediction mode or Deblocking Filter mode. In this mode, the one/four vectors decision is indicated by the MCBPC codeword for each macroblock. If only one motion vector is transmitted for a certain macroblock, this is defined as four vectors with the same value. If MCBPC indicates that four motion vectors are transmitted for the current macroblock, the information for the first motion vector is transmitted as the codeword MVD and the information for the three additional motion vectors is transmitted as the codewords $MVD_{2\text{-}4}$ (see also 5.3.7 and 5.3.8).

The vectors are obtained by adding predictors to the vector differences indicated by MVD and $MVD_{2\text{-}4}$ in a similar way as when only one motion vector per macroblock is present, according to the decision rules given in 6.1.1. Again the predictors are calculated separately for the horizontal and vertical components. However, the candidate predictors MV1, MV2 and MV3 are redefined as indicated in Figure F.1. If only one vector per macroblock is present, MV1, MV2 and MV3 are defined as for the 8 * 8 block numbered 1 in Figure 5 (this definition is given in the upper left of the four sub-figures of Figure F.1).

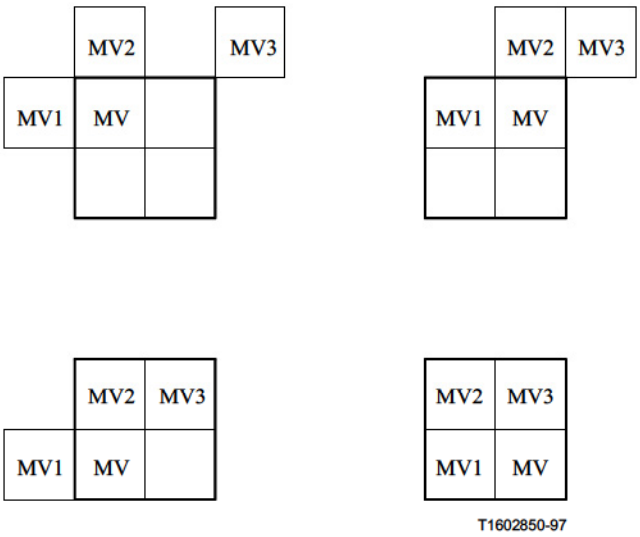



**Figure F.1/H.263 – Redefinition of the candidate predictors MV1, MV2 and MV3 for each of the luminance blocks in a macroblock**

If four vectors are used, each of the motion vectors is used for all pixels in one of the four luminance blocks in the macroblock. The numbering of the motion vectors is equivalent to the numbering of the four luminance blocks as given in Figure 5. Motion vector $MVD_{CHR}$ for both chrominance blocks is derived by calculating the sum of the four luminance vectors and dividing this sum by 8; the component values of the resulting sixteenth pixel resolution vectors are modified towards the nearest half-pixel position as indicated in Table F.1.

**Table F.1/H.263 – Modification of sixteenth pixel resolution chrominance vector components**

| **Sixteenth pixel position** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | /16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Resulting position** | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | /2 |

Half-pixel values are found using bilinear interpolation as described in 6.1.2. In Advanced Prediction mode, the prediction for luminance is obtained by overlapped motion compensation as described in F.3. The prediction for chrominance is obtained by applying the motion vector $MVD_{CHR}$ to all pixels in the two chrominance blocks (as it is done in the default prediction mode).

## F.3 Overlapped motion compensation for luminance

Each pixel in an 8 * 8 luminance prediction block is a weighted sum of three prediction values, divided by 8 (with rounding). In order to obtain the three prediction values, three motion vectors are used: the motion vector of the current luminance block, and two out of four "remote" vectors:

- the motion vector of the block at the left or right side of the current luminance block;
- the motion vector of the block above or below the current luminance block.

For each pixel, the remote motion vectors of the blocks at the two nearest block borders are used. This means that for the upper half of the block the motion vector corresponding to the block above the current block is used, while for the lower half of the block the motion vector corresponding to the block below the current block is used (see Figure F.3). Similarly, for the left half of the block the motion vector corresponding to the block at the left side of the current block is used, while for the

right half of the block the motion vector corresponding to the block at the right side of the current block is used (see Figure F.4).

Let *(x,y)* be a position in a picture measured in integer pixel units.

Let *(m,n)* be an integer block index in a picture, as given by:

$$m = x / 8 \qquad \text{and} \qquad n = y / 8,$$

where "/" denotes division with truncation.

Let *(i,j)* be an integer pixel location in an $8 \times 8$ block, given by:

$$i = x - m \cdot 8 \qquad \text{and} \qquad j = y - n \cdot 8,$$

resulting in:

$$(x,y) = (m \cdot 8 + i, n \cdot 8 + j).$$

Let $(MV^k{}_x, MV^k{}_y)$ be a motion vector, which may contain full-pixel or half pixel offsets, with $k = 0$, 1, or 2. For example, $(MV^k{}_x, MV^k{}_y)$ can be equal to (–7.0, 13.5). Here, $(MV^0{}_x, MV^0{}_y)$ denotes the motion vector for the current block *(m,n)*, $(MV^1{}_x, MV^1{}_y)$ denotes the motion vector of the block either above or below, and $(MV^2{}_x, MV^2{}_y)$ denotes the motion vector either to the left or right of the current block *(m,n)* as defined above.

Then the creation of each pixel, *P(x,y),* in an $8 \times 8$ luminance prediction block with block index *(m,n)* is governed by the following equation:

$$P(x,y) = (q(x,y) \cdot H_0(i,j) + r(x,y) \cdot H_1(i,j) + s(x,y) \cdot H_2(i,j) + 4)/8,$$

where *q(x,y), r(x,y)* and *s(x,y)* are the prediction values taken from the referenced picture as defined by:

$$q(x,y) = p(x + MV^0{}_x, y + MV^0{}_y),$$

$$r(x,y) = p(x + MV^1{}_x, y + MV^1{}_y),$$

$$s(x,y) = p(x + MV^2{}_x, y + MV^2{}_y),$$

where $p(x + MV^k{}_x, y + MV^k{}_y)$ is the prediction value at position $(x + MV^k{}_x, y + MV^k{}_y)$ in the referenced picture. Note that $(x + MV^k{}_x, y + MV^k{}_y)$ may be outside the picture and can be at a full or half-pixel position. In cases using half-pixel motion vectors, $p(x + MV^k{}_x, y + MV^k{}_y)$ refers to the value obtained after the interpolation process described in 6.1.2 is applied.

The matrices $H_0(i,j)$, $H_1(i,j)$ and $H_2(i,j)$ are defined in Figures F.2, F.3 and F.4, where *(i, j)* denotes the column and row, respectively, of the matrix.

When neither the Slice Structured mode (see Annex K) nor the Independent Segment Decoding mode (see Annex R) are in use, remote motion vectors from other video picture segments are used in the same way as remote motion vectors inside the current GOB. If either the Slice Structured mode or the Independent Segment Decoding mode are in use, the remote motion vectors corresponding to blocks from other video picture segments are set to the motion vector of the current block, regardless of the other conditions described in the next paragraph. (See Annex R for the definition of a video picture segment.)

If one of the surrounding macroblocks was not coded, the corresponding remote motion vector is set to zero. If one of the surrounding blocks was INTRA coded, the corresponding remote motion vector is replaced by the motion vector for the current block except when in PB-frames mode. In this case (INTRA block in PB-frame mode), the INTRA block's motion vector is used (also see Annex G). If the current block is at the border of the picture and therefore a surrounding block is not present, the corresponding remote motion vector is replaced by the current motion vector. In all cases, if the current block is at the bottom of the macroblock (for block number 3 or 4, see Figure 5), the remote motion vector corresponding with an 8 * 8 luminance block in the macroblock below the current macroblock is replaced by the motion vector for the current block.

The weighting values for the prediction are given in Figures F.2, F.3 and F.4.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | 5 | 5 | 5 | 5 | 5 | 5 | 4 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 5 | 5 | 6 | 6 | 6 | 6 | 5 | 5 |
| 5 | 5 | 6 | 6 | 6 | 6 | 5 | 5 |
| 5 | 5 | 6 | 6 | 6 | 6 | 5 | 5 |
| 5 | 5 | 6 | 6 | 6 | 6 | 5 | 5 |
| 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 4 | 5 | 5 | 5 | 5 | 5 | 5 | 4 |

**Figure F.2/H.263 – Weighting values, $H_0$, for prediction with motion vector of current luminance block**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 1 | 1 | 2 | 2 | 2 | 2 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 2 | 2 | 2 | 2 | 1 | 1 |
| 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |

**Figure F.3/H.263 – Weighting values, $H_1$, for prediction with motion vectors of the luminance blocks on top or bottom of current luminance block**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 2 |
| 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 |
| 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 |
| 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 |
| 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 |
| 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 |
| 2 | 2 | 1 | 1 | 1 | 1 | 2 | 2 |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 2 |

**Figure F.4/H.263 – Weighting values, $H_2$, for prediction with motion vectors of the luminance blocks to the left or right of current luminance block**

# ANNEX G

## PB-frames mode

### G.1 Introduction

This annex describes the optional PB-frames mode of this Recommendation. The capability of this mode is signalled by external means (for example, Recommendation H.245). The use of this mode is indicated in PTYPE.

A PB-frame consists of two pictures being coded as one unit. The name PB comes from the name of picture types in Recommendation H.262 where there are P-pictures and B-pictures. Thus, a PB-frame consists of one P-picture which is predicted from the previous decoded P-picture and one B-picture which is predicted both from the previous decoded P-picture and the P-picture currently being decoded. The name B-picture was chosen because parts of B-pictures may be bidirectionally predicted from the past and future pictures. The prediction process is illustrated in Figure G.1.

An improved version of the PB-frames mode, termed the "Improved PB-frames mode," is described in Annex M. The PB-frames mode as described in this annex is retained herein only for purposes of compatibility with systems designed prior to the adoption of the Improved PB-frames mode. For this reason, the PB-frames mode as described in this annex cannot be used with the additional features of the syntax which require the use of PLUSPTYPE.

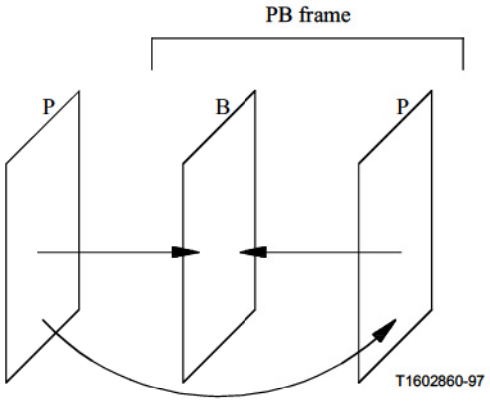



**Figure G.1/H.263 – Prediction in PB-frames mode**

## G.2 PB-frames and INTRA blocks

When PB-frames are used, the coding mode INTRA has the following meaning (see also 5.3.2):

- The P-blocks are INTRA coded.
- The B-blocks are INTER coded with prediction as for an INTER block.

If PB-frames are used, Motion Vector Data (MVD) is included also for INTRA macroblocks in pictures for which PTYPE indicates "INTER". In this case, the vector is used for the B-blocks only. The codewords $MVD_{2\text{-}4}$ are never used for INTRA (see also Table 10). When in both the Advanced Prediction mode and the PB-frames mode, and one of the surrounding blocks was coded in INTRA mode, the corresponding remote motion vector is not replaced by the motion vector for the current block. Instead, the remote "INTRA" motion vector is used.

## G.3 Block layer

In a PB-frame, a macroblock comprises twelve blocks. First, the data for the six P-blocks is transmitted as in the default H.263 mode, then the data for the six B-blocks (see also 5.4). The structure of the block layer is shown in Figure 11. INTRADC is present for every P-block of the macroblock if MCBPC indicates macroblock type 3 or 4 (see Tables 7 and 8). INTRADC is not present for B-blocks. TCOEF is present for P-blocks if indicated by MCBPC or CBPY; TCOEF is present for B-blocks if indicated by CBPB.

## G.4 Calculation of vectors for the B-picture in a PB-frame

The vectors for the B-picture are calculated as follows (see also 6.1.1). Assume we have a vector component MV in half-pixel units to be used in the P-picture (MV represents a vector component for an 8 * 8 luminance block; if only one vector per macroblock is transmitted, MV has the same value for each of the four 8 * 8 luminance blocks). For prediction of the B-picture we need both forward and backward vector components $MV_F$ and $MV_B$. These forward and backward vector components are derived from MV and eventually enhanced by a delta vector given by MVDB.

- $TR_D$: The increment of the temporal reference TR (or the combined extended temporal reference ETR and temporal reference TR in an Improved PB frame when a custom picture clock frequency is in use) from the last picture header (see 5.1.2). If $TR_D$ is negative, then $TR_D = TR_D + d$ where d = 256 for CIF picture frequency and 1024 for any custom picture clock frequency.
- $TR_B$: See 5.1.2.

Assume that $MV_D$ is the delta vector component given by MVDB and corresponding with vector component MV. If MVDB is not present, $MV_D$ is set to zero. If MVDB is present, the same $MV_D$ given by MVDB is used for each of the four luminance B-blocks within the macroblock.

Now, $MV_F$ and $MV_B$ are given in half-pixel units by the following formulas:

$$MV_F = (TR_B \times MV) \;/\; TR_D + MV_D$$

$$MV_B = ((TR_B - TR_D) \times MV) \;/\; TR_D \qquad \text{if } MV_D \text{ is equal to 0}$$

$$MV_B = MV_F - MV \qquad \text{if } MV_D \text{ is unequal to 0}$$

where "/" means division by truncation. It is assumed that the scaling reflects the actual position in time of P- and B-pictures. Advantage is taken of the fact that the range of values for $MV_F$ is constrained. Each VLC word for MVDB represents a pair of difference values. Only one of the pair will yield a value for $MV_F$ falling within the permitted range (default [−16, 15.5]; in Unrestricted Motion Vector mode [−31.5, 31.5]). The formulas for $MV_F$ and $MV_B$ are also used in the case of INTRA blocks where the vector data is used only for predicting B-blocks.

For chrominance blocks, $MV_F$ is derived by calculating the sum of the four corresponding luminance $MV_F$ vectors and dividing this sum by 8; the resulting sixteenth pixel resolution vector components are modified towards the nearest half-pixel position as indicated in Table F.1. $MV_B$ for chrominance is derived by calculating the sum of the four corresponding luminance $MV_B$ vectors and dividing this sum by 8; the resulting sixteenth pixel resolution vector components are modified towards the nearest half-pixel position as indicated in Table F.1.

A positive value of the horizontal or vertical component of the motion vector signifies that the prediction is formed from pixels in the referenced picture which are spatially to the right or below the pixels being predicted.

## G.5 Prediction of a B-block in a PB-frame

In this subclause a block means an 8 × 8 block. The following procedure applies for luminance as well as chrominance blocks. First, the forward and backward vectors are calculated. It is assumed that the P-macroblock (luminance and chrominance) is first decoded, reconstructed and clipped (see 6.3.2). This macroblock is called $P_{REC}$. Based on $P_{REC}$ and the prediction for $P_{REC}$, the prediction for the B-block is calculated.

The prediction of the B-block has two modes that are used for different parts of the block:

- For pixels where the backward vector – $MV_B$ – points inside $P_{REC}$, use bidirectional prediction. This is obtained as the average of the forward prediction using $MV_F$ relative to the previous decoded picture, and the backward prediction using $MV_B$ relative to $P_{REC}$. The average is calculated by dividing the sum of the two predictions by two (division by truncation).
- For all other pixels, forward prediction using $MV_F$ relative to the previous decoded picture is used.

Figure G.2 indicates which part of a block is predicted bidirectionally (shaded part of the B-block) and which part with forward prediction only (rest of the B-block).

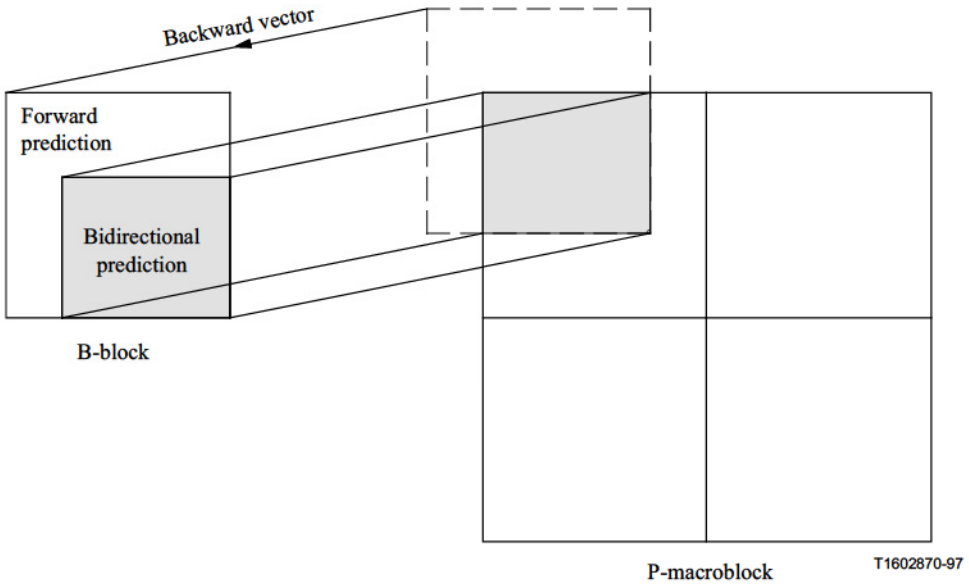



**Figure G.2/H.263 – Forward and bidirectional prediction for a B-block**

Bidirectional prediction is used for pixels where the backward vector – $MV_B$ – points inside $P_{REC}$. These pixels are defined by the following procedures which are specified in C language.

Definitions:

| | |
|---|---|
| nh | Horizontal position of block within a macroblock (0 or 1). |
| nv | Vertical position of block within a macroblock (0 or 1). |
| mh(nh,nv) | Horizontal vector component of block (nh,nv) in half-pixel units. |
| mv(nh,nv) | Vertical vector component of block (nh,nv) in half-pixel units. |
| mhc | Horizontal chrominance vector component. |
| mvc | Vertical chrominance vector component. |

**Procedure for luminance**

```
for (nh = 0; nh <= 1; nh++) {
      for (nv = 0; nv <= 1; nv++) {
         for (i = nh * 8 + max(0,(–mh(nh,nv)+1)/2 – nh * 8);
               i <= nh * 8 + min(7,15–(mh(nh,nv)+1)/2 – nh * 8); i++) {
            for (j = nv * 8 + max(0,(–mv(nh,nv)+1)/2 – nv * 8);
               j <= nv * 8 + min(7,15–(mv(nh,nv)+1)/2 – nv * 8); j++) {
               predict pixel (i,j) bidirectionally
            }
         }
      }
}
```

**Procedure for chrominance**

```
for (i = max(0, (–mhc+1)/2); i <= min(7,7–(mhc+1)/2); i++) {
      for (j = max(0, (–mvc+1)/2); j <= min(7,7–(mvc+1)/2); j++) {
         predict pixel (i,j) bidirectionally;
      }
}
```

Pixels not predicted bidirectionally are predicted with forward prediction only.

# ANNEX H

## Forward error correction for coded video signal

### H.1 Introduction

This annex describes an optional forward error correction method (code and framing) for transmission of H.263 encoded video data. This forward error correction may be used in situations where no forward error correction is provided by external means, for example at the multiplex or system level. It is not used for Recommendation H.324. Both the framing and the forward error correction code are the same as in Recommendation H.261.

### H.2 Error correction framing

To allow the video data and error correction parity information to be identified by a decoder, an error correction framing pattern is included. This pattern consists of multiframes of eight frames, each frame comprising 1 bit framing, 1 bit fill indicator (Fi), 492 bits of coded data (or fill all 1s) and 18 bits parity (see Figure H.1). For each multiframe the frame alignment pattern formed by the framing bits of the eight individual frames is:

$$(S_1S_2S_3S_4S_5S_6S_7S_8) = (00011011)$$

The fill indicator (Fi) can be set to zero by an encoder. In this case 492 consecutive fill bits (fill all 1s) are used instead of 492 bits of coded data. This may be used for stuffing data (see 3.6).

### H.3 Error correcting code

The error correction code is a BCH (511, 493) forward error correction code. Use of this by the decoder is optional. The parity is calculated against a code of 493 bits, comprising 1 bit fill indicator (Fi) and 492 bits of coded video data.

The generator polynomial is:

$$g(x) = (x9 + x4 + 1)(x9 + x6 + x4 + x3 + 1)$$

Example: for the input data of "01111 ... 11" (493 bits) the resulting correction parity bits are "011011010100011011" (18 bits).

### H.4 Relock time for error corrector framing

Three consecutive error correction frame alignment patterns (24 bits) should be received before frame lock is deemed to have been achieved. The decoder should be designed such that frame lock will be re-established within 34 000 bits after an error corrector framing phase change.

NOTE – This assumes that the video data does not contain three correctly phased emulations of the error correction framing sequence during the relocking period.

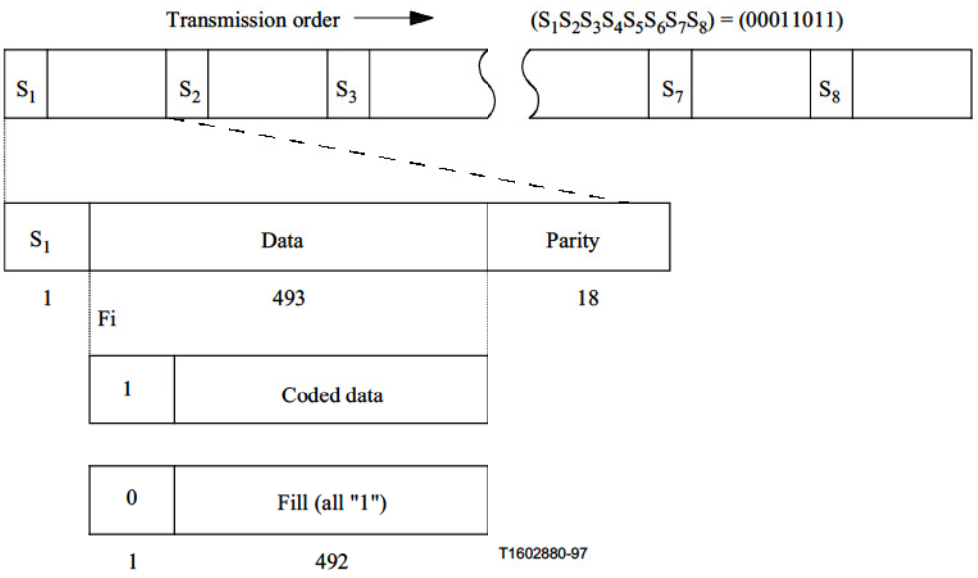



**Figure H.1/H.263 – Error correcting frame**

ANNEX I

## Advanced INTRA Coding mode

This annex describes the optional Advanced INTRA Coding mode of this Recommendation. The capability of this H.263 mode is signalled by external means (for example, Recommendation H.245). The use of this mode is indicated in the PLUSPTYPE field of the picture header.

### I.1 Introduction

This optional mode alters the decoding of macroblocks of type "INTRA" (macroblocks of other types are not affected). The coding efficiency of INTRA macroblocks is improved by using:

1) INTRA-block prediction using neighbouring INTRA blocks for the same component (Y, $C_B$, or $C_R$);

2) modified inverse quantization for INTRA coefficients; and

3) a separate VLC for INTRA coefficients.

A particular INTRA-coded block may be predicted from the block above the current block being decoded, from the block to the left of the current block being decoded, or from both. Special cases exist for situations in which the neighbouring blocks are not INTRA coded or are not in the same video picture segment. Block prediction always uses data from the same luminance or color difference component (Y, $C_B$, or $C_R$) as the block being decoded. In prediction, DC coefficients are always predicted in some manner. The first row of AC coefficients may be predicted from those in the block above, or the first column of AC coefficients may be predicted from those in the block to the left, or only the DC coefficient may be predicted as an average from the block above and the block to the left, as signalled on a macroblock-by-macroblock basis. The remaining AC coefficients are never predicted. Inverse quantization of the INTRADC coefficient is modified to allow a varying quantization step size, unlike in the main text of this Recommendation where a fixed step size of 8 is used for INTRADC coefficients. Inverse quantization of all INTRA coefficients is performed without a "dead-zone" in the quantizer reconstruction spacing.

## I.2 Syntax

When using the Advanced INTRA Coding mode, the macroblock layer syntax is altered as specified in Figure I.1. The syntax as shown in Figure I.1 is the same as that defined in 5.3 except for the insertion of an additional INTRA_MODE field for INTRA macroblocks. INTRA_MODE is present only when MCBPC indicates a macroblock of type INTRA (macroblock type 3 or 4). The prediction mode is coded using the variable length code shown in Table I.1. One prediction mode is transmitted per INTRA macroblock.

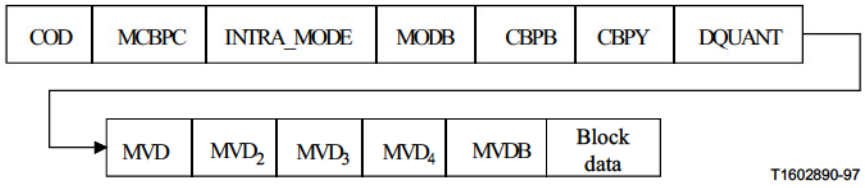



**Figure I.1/H.263 – Structure of macroblock layer**

**Table I.1/H.263 – VLC for INTRA_MODE**

| Index | Prediction Mode | VLC |
|---|---|---|
| 0 | 0 (DC Only) | 0 |
| 1 | 1 (Vertical DC & AC) | 10 |
| 2 | 2 (Horizontal DC & AC) | 11 |

## I.3 Decoding process

Two scans in addition to the zigzag scan are employed. The two added scans are shown in Figure I.2 parts a) and b), and the zigzag scan is shown in Figure 14.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 11 | 12 | 13 | 14 |
| 5 | 6 | 9 | 10 | 18 | 17 | 16 | 15 |
| 7 | 8 | 20 | 19 | 27 | 28 | 29 | 30 |
| 21 | 22 | 25 | 26 | 31 | 32 | 33 | 34 |
| 23 | 24 | 35 | 36 | 43 | 44 | 45 | 46 |
| 37 | 38 | 41 | 42 | 47 | 48 | 49 | 50 |
| 39 | 40 | 51 | 52 | 57 | 58 | 59 | 60 |
| 53 | 54 | 55 | 56 | 61 | 62 | 63 | 64 |

**a) Alternate-Horizontal scan**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 5 | 7 | 21 | 23 | 37 | 39 | 53 |
| 2 | 6 | 8 | 22 | 24 | 38 | 40 | 54 |
| 3 | 9 | 20 | 25 | 35 | 41 | 51 | 55 |
| 4 | 10 | 19 | 26 | 36 | 42 | 52 | 56 |
| 11 | 18 | 27 | 31 | 43 | 47 | 57 | 61 |
| 12 | 17 | 28 | 32 | 44 | 48 | 58 | 62 |
| 13 | 16 | 29 | 33 | 45 | 49 | 59 | 63 |
| 14 | 15 | 30 | 34 | 46 | 50 | 60 | 64 |

**b) Alternate-Vertical scan (as in Recommendation H.262)**

**Figure I.2/H.263 – Alternate DCT scanning patterns for Advanced INTRA coding**

For INTRA-coded blocks, if Prediction Mode = 0, the zigzag scan shown in Figure 14 is selected for all blocks in the macroblock, otherwise, the prediction direction is used to select a scan for the macroblock.

Prediction Mode = 1 uses the vertically adjacent block in forming a prediction. This prediction mode is designed for INTRA blocks which are dominated by stronger horizontal frequency content, so the vertically adjacent block is used to predict the horizontal frequency content of the current block, with a prediction of zero for all coefficients representing vertical AC content. Then the scanning pattern is chosen to scan the stronger horizontal frequencies prior to the vertical ones, using the Alternate-Horizontal scan.

Prediction Mode = 2 uses the horizontally adjacent block in forming a prediction. This prediction mode is designed for INTRA blocks which are dominated by stronger vertical frequency content, so the horizontally adjacent block is used to predict the vertical frequency content of the current block, with a prediction of zero for all coefficients representing horizontal AC content. Then the scanning pattern is chosen to scan the stronger vertical frequencies prior to the horizontal ones, using the Alternate-Vertical scan.

For non-INTRA blocks, the $8 \times 8$ blocks of transform coefficients are scanned with "zigzag" scanning as shown in Figure 14.

A separate VLC table is used for all INTRADC and INTRA AC coefficients. This table is specified in Table I.2. Note that the VLC codeword entries used in Table I.2 are the same as those used in the normal TCOEF table (Table 16) used when Advanced INTRA coding is not in use, but with a different interpretation of LEVEL and RUN (without altering LAST).

**Table I.2/H.263 – VLC for INTRA TCOEF**

| INDEX | LAST | RUN | \|LEVEL\| | BITS | VLC CODE |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 3 | 10s |
| 1 | 0 | 1 | 1 | 5 | 1111s |
| 2 | 0 | 3 | 1 | 7 | 0101 01s |
| 3 | 0 | 5 | 1 | 8 | 0010 111s |
| 4 | 0 | 7 | 1 | 9 | 0001 1111s |
| 5 | 0 | 8 | 1 | 10 | 0001 0010 1s |
| 6 | 0 | 9 | 1 | 10 | 0001 0010 0s |
| 7 | 0 | 10 | 1 | 11 | 0000 1000 01s |
| 8 | 0 | 11 | 1 | 11 | 0000 1000 00s |
| 9 | 0 | 4 | 3 | 12 | 0000 0000 111s |
| 10 | 0 | 9 | 2 | 12 | 0000 0000 110s |
| 11 | 0 | 13 | 1 | 12 | 0000 0100 000s |
| 12 | 0 | 0 | 2 | 4 | 110s |
| 13 | 0 | 1 | 2 | 7 | 0101 00s |
| 14 | 0 | 1 | 4 | 9 | 0001 1110s |
| 15 | 0 | 1 | 5 | 11 | 0000 0011 11s |
| 16 | 0 | 1 | 6 | 12 | 0000 0100 001s |
| 17 | 0 | 1 | 7 | 13 | 0000 0101 0000s |
| 18 | 0 | 0 | 3 | 5 | 1110s |
| 19 | 0 | 3 | 2 | 9 | 0001 1101s |
| 20 | 0 | 2 | 3 | 11 | 0000 0011 10s |
| 21 | 0 | 3 | 4 | 13 | 0000 0101 0001s |
| 22 | 0 | 0 | 5 | 6 | 0110 1s |
| 23 | 0 | 4 | 2 | 10 | 0001 0001 1s |
| 24 | 0 | 3 | 3 | 11 | 0000 0011 01s |
| 25 | 0 | 0 | 4 | 6 | 0110 0s |

**Table I.2/H.263 – VLC for INTRA TCOEF** (*continued*)

| INDEX | LAST | RUN | \|LEVEL\| | BITS | VLC CODE |
|---|---|---|---|---|---|
| 26 | 0 | 5 | 2 | 10 | 0001 0001 0s |
| 27 | 0 | 5 | 3 | 13 | 0000 0101 0010s |
| 28 | 0 | 2 | 1 | 6 | 0101 1s |
| 29 | 0 | 6 | 2 | 11 | 0000 0011 00s |
| 30 | 0 | 0 | 25 | 13 | 0000 0101 0011s |
| 31 | 0 | 4 | 1 | 7 | 0100 11s |
| 32 | 0 | 7 | 2 | 11 | 0000 0010 11s |
| 33 | 0 | 0 | 24 | 13 | 0000 0101 0100s |
| 34 | 0 | 0 | 8 | 7 | 0100 10s |
| 35 | 0 | 8 | 2 | 11 | 0000 0010 10s |
| 36 | 0 | 0 | 7 | 7 | 0100 01s |
| 37 | 0 | 2 | 4 | 11 | 0000 0010 01s |
| 38 | 0 | 0 | 6 | 7 | 0100 00s |
| 39 | 0 | 12 | 1 | 11 | 0000 0010 00s |
| 40 | 0 | 0 | 9 | 8 | 0010 110s |
| 41 | 0 | 0 | 23 | 13 | 0000 0101 0101s |
| 42 | 0 | 2 | 2 | 8 | 0010 101s |
| 43 | 0 | 1 | 3 | 8 | 0010 100s |
| 44 | 0 | 6 | 1 | 9 | 0001 1100s |
| 45 | 0 | 0 | 10 | 9 | 0001 1011s |
| 46 | 0 | 0 | 12 | 10 | 0001 0000 1s |
| 47 | 0 | 0 | 11 | 10 | 0001 0000 0s |
| 48 | 0 | 0 | 18 | 10 | 0000 1111 1s |
| 49 | 0 | 0 | 17 | 10 | 0000 1111 0s |
| 50 | 0 | 0 | 16 | 10 | 0000 1110 1s |
| 51 | 0 | 0 | 15 | 10 | 0000 1110 0s |
| 52 | 0 | 0 | 14 | 10 | 0000 1101 1s |
| 53 | 0 | 0 | 13 | 10 | 0000 1101 0s |
| 54 | 0 | 0 | 20 | 12 | 0000 0100 010s |
| 55 | 0 | 0 | 19 | 12 | 0000 0100 011s |
| 56 | 0 | 0 | 22 | 13 | 0000 0101 0110s |
| 57 | 0 | 0 | 21 | 13 | 0000 0101 0111s |
| 58 | 1 | 0 | 1 | 5 | 0111s |
| 59 | 1 | 14 | 1 | 10 | 0000 1100 1s |
| 60 | 1 | 20 | 1 | 12 | 0000 0000 101s |
| 61 | 1 | 1 | 1 | 7 | 0011 11s |
| 62 | 1 | 19 | 1 | 12 | 0000 0000 100s |
| 63 | 1 | 2 | 1 | 7 | 0011 10s |
| 64 | 1 | 3 | 1 | 7 | 0011 01s |
| 65 | 1 | 0 | 2 | 7 | 0011 00s |
| 66 | 1 | 5 | 1 | 8 | 0010 011s |
| 67 | 1 | 6 | 1 | 8 | 0010 010s |
| 68 | 1 | 4 | 1 | 8 | 0010 001s |
| 69 | 1 | 0 | 3 | 8 | 0010 000s |
| 70 | 1 | 9 | 1 | 9 | 0001 1010s |
| 71 | 1 | 10 | 1 | 9 | 0001 1001s |
| 72 | 1 | 11 | 1 | 9 | 0001 1000s |
| 73 | 1 | 12 | 1 | 9 | 0001 0111s |
| 74 | 1 | 13 | 1 | 9 | 0001 0110s |

**Table I.2/H.263 – VLC for INTRA TCOEF** (*concluded*)

| INDEX | LAST | RUN | \|LEVEL\| | BITS | VLC CODE |
|---|---|---|---|---|---|
| 75 | 1 | 8 | 1 | 9 | 0001 0101s |
| 76 | 1 | 7 | 1 | 9 | 0001 0100s |
| 77 | 1 | 0 | 4 | 9 | 0001 0011s |
| 78 | 1 | 17 | 1 | 10 | 0000 1100 0s |
| 79 | 1 | 18 | 1 | 10 | 0000 1011 1s |
| 80 | 1 | 16 | 1 | 10 | 0000 1011 0s |
| 81 | 1 | 15 | 1 | 10 | 0000 1010 1s |
| 82 | 1 | 2 | 2 | 10 | 0000 1010 0s |
| 83 | 1 | 1 | 2 | 10 | 0000 1001 1s |
| 84 | 1 | 0 | 6 | 10 | 0000 1001 0s |
| 85 | 1 | 0 | 5 | 10 | 0000 1000 1s |
| 86 | 1 | 4 | 2 | 11 | 0000 0001 11s |
| 87 | 1 | 3 | 2 | 11 | 0000 0001 10s |
| 88 | 1 | 1 | 3 | 11 | 0000 0001 01s |
| 89 | 1 | 0 | 7 | 11 | 0000 0001 00s |
| 90 | 1 | 2 | 3 | 12 | 0000 0100 100s |
| 91 | 1 | 1 | 4 | 12 | 0000 0100 101s |
| 92 | 1 | 0 | 9 | 12 | 0000 0100 110s |
| 93 | 1 | 0 | 8 | 12 | 0000 0100 111s |
| 94 | 1 | 21 | 1 | 13 | 0000 0101 1000s |
| 95 | 1 | 22 | 1 | 13 | 0000 0101 1001s |
| 96 | 1 | 23 | 1 | 13 | 0000 0101 1010s |
| 97 | 1 | 7 | 2 | 13 | 0000 0101 1011s |
| 98 | 1 | 6 | 2 | 13 | 0000 0101 1100s |
| 99 | 1 | 5 | 2 | 13 | 0000 0101 1101s |
| 100 | 1 | 3 | 3 | 13 | 0000 0101 1110s |
| 101 | 1 | 0 | 10 | 13 | 0000 0101 1111s |
| 102 | ESCAPE | | | 7 | 0000 011 |

Depending on the value of INTRA_MODE, either one or eight coefficients are prediction residuals that must be added to a predictor as described below. Figure I.3 shows three 8 × 8 blocks of final reconstructed DCT levels for the same component (Y, $C_B$, or $C_R$), labelled RecA'(u,v), RecB'(u,v) and RecC'(u,v), where u and v are column (horizontal) and row (vertical) indices, respectively. The reconstruction process differs from the processing described in 6.2.1. INTRADC residuals are reconstructed differently by use of a variable step size, as opposed to using Table 15, and then a predictor is added to the residual values to obtain the final coefficient reconstruction value. INTRA coefficients other than INTRADC are also reconstructed differently than in 6.2.1, using a reconstruction spacing without a "dead-zone" and then in some cases adding a predictor to obtain the final coefficient reconstruction value. The block may contain both DC and AC prediction residuals.

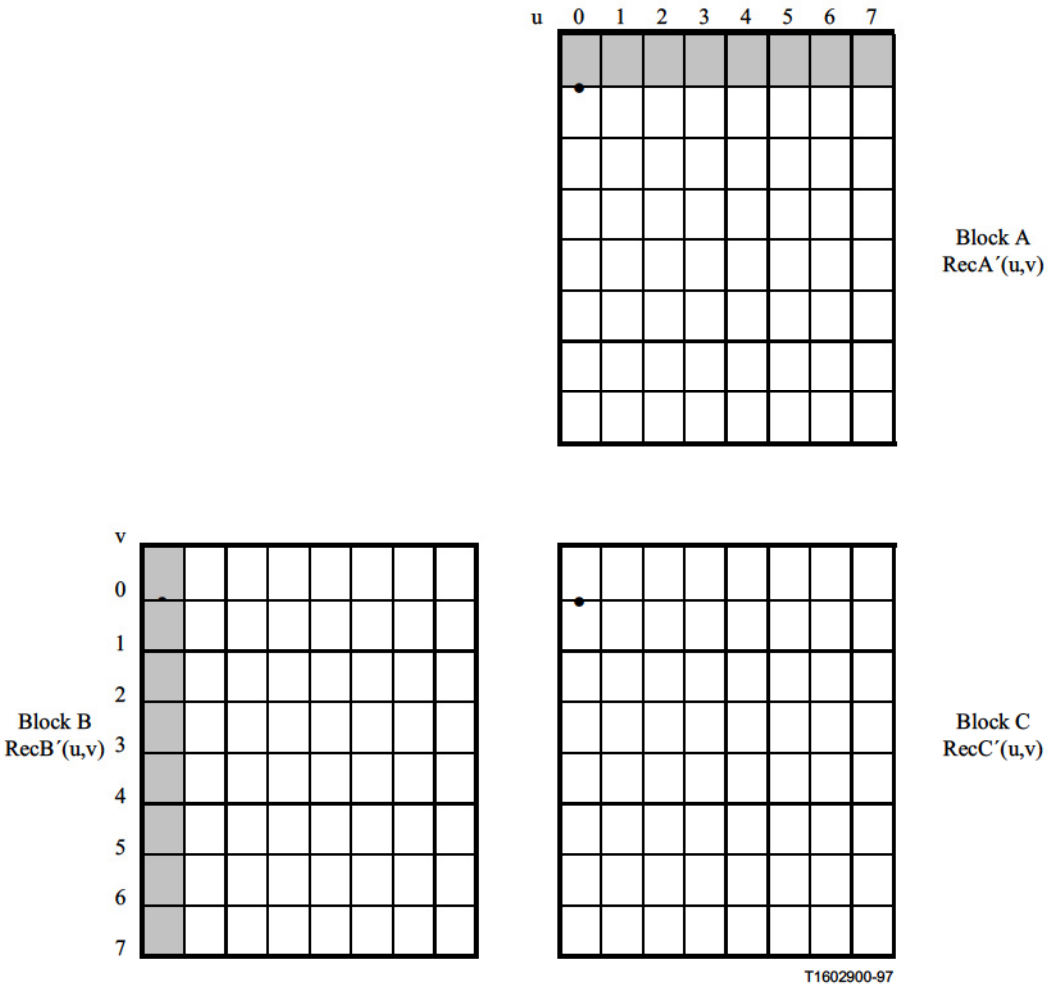



**Figure I.3/H.263 – Three neighbouring blocks in the DCT domain**

The definitions of MCBPC and CBPY are changed when Advanced INTRA coding is in use. When Advanced INTRA coding is in use, INTRADC transform coefficients are no longer handled as a separate case, but are instead treated in the same way as the AC coefficients in regard to MCBPC and CBPY. This means that a zero INTRADC will not be coded as a LEVEL, but will simply increase the run for the following AC coefficients.

The inverse quantization process for the B-part of an Improved PB frame (see Annex M) is not altered by the use of the Advanced INTRA coding mode.

Define RecC(u,v) to be the reconstructed coefficient *residuals* of the current block. For all INTRA coefficients, the reconstructed residual value is obtained by:

$$RecC(u,v) = 2 * QUANT * LEVEL(u,v) \qquad u = 0, \ldots, 7, v = 0, \ldots, 7.$$

NOTE – LEVEL(u,v) denotes a quantity having both a magnitude and a sign in the above equation.

Define RecC′(u,v) to be the *final* reconstructed coefficient values of the current block (after adjustments for prediction, oddification as described below, and clipping). The final reconstructed coefficient values RecC′(u,v) are recovered by adding RecC(u,v) to the appropriate prediction as signalled in the INTRA_MODE field, altering the least-significant bit if needed for oddification of the DC coefficient, and clipping.