INTERNATIONAL STANDARD

ISO/IEC 23002-4

First edition
2010-01-15

Information technology — MPEG video technologies —

Part 4:
Video tool library

Technologies de l'information — Technologies vidéo MPEG —

Partie 4: Bibliothèque d'outils vidéo

Reference number
ISO/IEC 23002-4:2010(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 23002-4 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

ISO/IEC 23002 consists of the following parts, under the general title *Information technology — MPEG video technologies*:

- *Part 1: Accuracy requirements for implementation of integer-output 8×8 inverse discrete cosine transform*
- *Part 2: Fixed-point 8×8 inverse discrete cosine transform and discrete cosine transform*
- *Part 3: Representation of auxiliary video and supplemental information*
- *Part 4: Video tool library*

Introduction

This part of ISO/IEC 23002 defines the MPEG video tool library, which contains tools drawn from existing MPEG coding standards, such as ISO/IEC 14496-2 and ISO/IEC 14496-10, and ISO/IEC 23001-4 defines the methods capable of describing codec configurations in the reconfigurable video coding (RVC) framework.

This part of ISO/IEC 23002 primarily addresses reconfigurable video aspects and will only focus on the description of representation of video codec configurations under the RVC framework, but could be extended to a more generic reconfigurable media coding (RMC) framework.

The objective of RVC is to offer a framework that is capable of configuring and specifying video codecs as a collection of “higher level” modules by using video coding tools. The video coding tools are defined in video tool libraries. This part of ISO/IEC 23002 defines the MPEG video tool library. The RVC framework principle could also support non-MPEG tool libraries, provided that their developers have taken care to obey the appropriate rules of operation.

For the purpose of framework deployment, an appropriate description is needed to describe configurations of decoders composed of or instantiated from a subset of video tools from either one or more libraries. As illustrated in Figure 1, the configuration information consists of

— bitstream syntax description, and

— network of functional units (FUs) description (also referred to as the decoder configuration)

that together constitute the entire decoder description.

Bitstreams of existing MPEG standards are specified by specific syntax structures and decoders are composed of various coding tools. Therefore, RVC includes support for bitstream syntax descriptions as well as video coding tools. As depicted in Figure 1, a typical RVC decoder requires two types of information, namely the decoder description and the encoded media (e.g. video bitstreams) data.

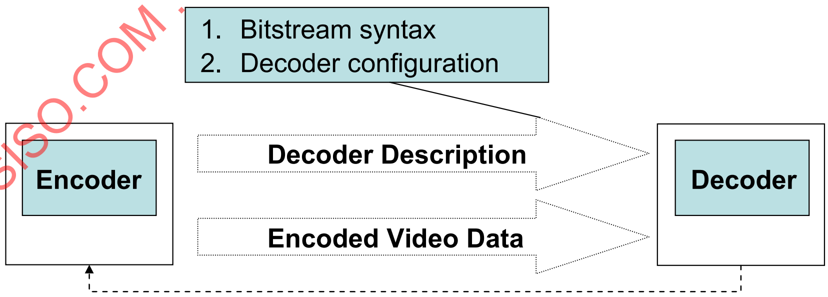


Figure 1 — Conceptual diagram of RVC

A more detailed description of the RVC decoder is illustrated in Figure 2. As shown in Figure 2, the decoder description is required for the configuration of a RVC decoder. The Bitstream Syntax Description (BSD) and FU Network Description (FND) (which compose the Decoder Description) are used to configure or compose an abstract decoder model (ADM) which is instantiated through the selection of FUs from tool libraries optionally with proper parameter assignment. Such ADM constitutes the behavioral reference model used in setting up a decoding solution under the RVC framework. The process of yielding a decoding solution may vary depending on the technologies used for the desired implementations. Examples of the instantiation of an ADM and generation of proprietary decoding solutions can be found in ISO/IEC 23001-4.

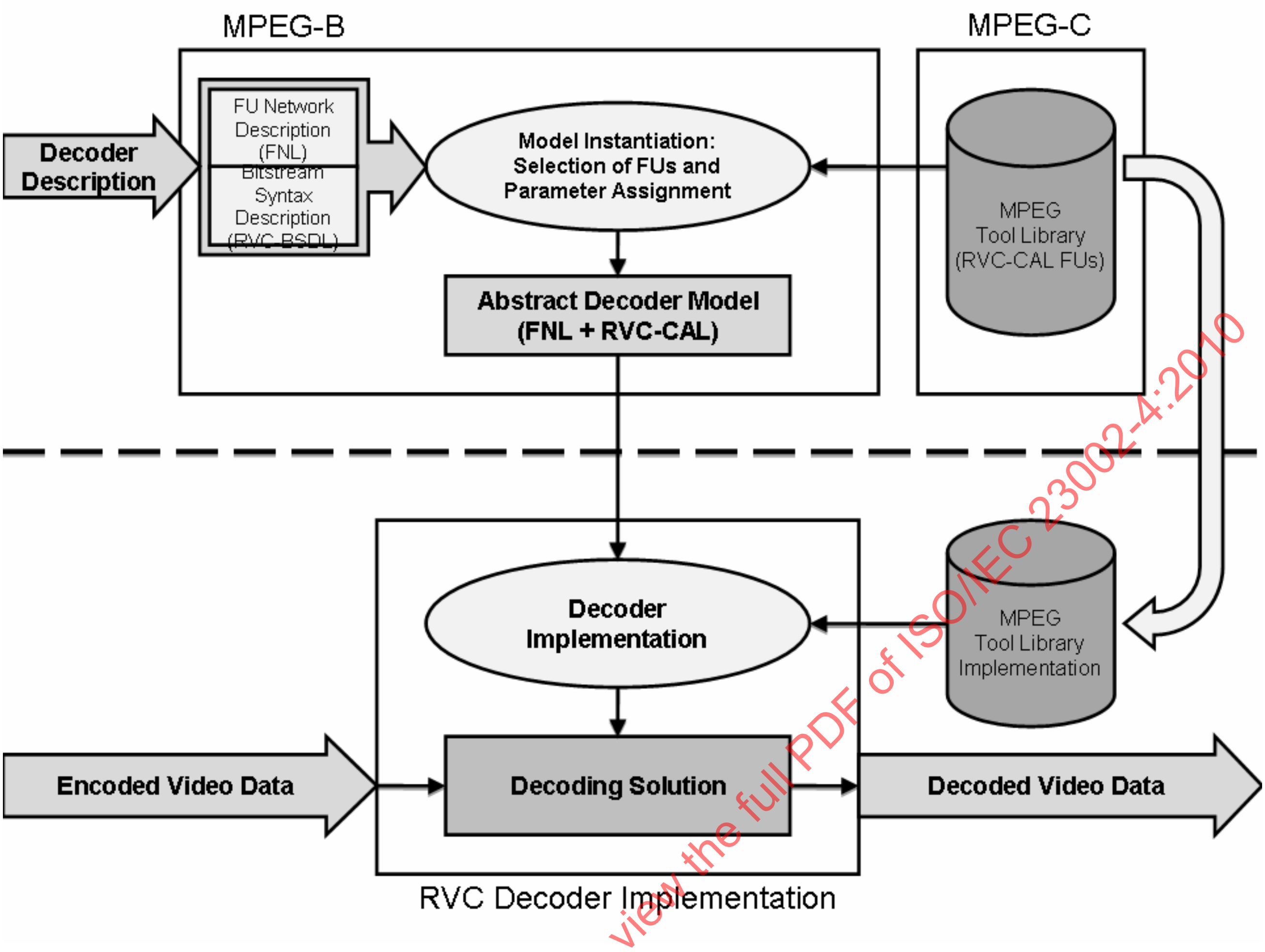

MPEG-B
MPEG-C
Decoder Description
FU Network Description (FNL)
Bitstream Syntax Description (RVC-BSDL)
Model Instantiation: Selection of FUs and Parameter Assignment
Abstract Decoder Model (FNL + RVC-CAL)
MPEG Tool Library (RVC-CAL FUs)
Decoder Implementation
MPEG Tool Library Implementation
Encoded Video Data
Decoding Solution
Decoded Video Data
RVC Decoder Implementation

Figure 2 — Graphical representation of the process for setting up a decoding solution under the RVC framework

Within the RVC framework, the decoder description describes a particular decoder configuration and consists of the FND and the BSD. The FND describes the connectivity of the network of FUs used to form a decoder whereas the parsing process for the bitstream syntax is implicitly described by the BSD. These two descriptions are specified using two standard XML-based languages or dialects:

— Functional unit network language (FNL) is a language that describes the FND, known also as "network of FUs". The FNL specified normatively within the scope of the RVC framework is provided in ISO/IEC 23001-4.

— Bitstream syntax description language (BSDL), standardized in ISO/IEC 23001-5 (MPEG-B Part 5), describes the bitstream syntax and the parsing rules. A pertinent subset of this BSDL named RVC-BSDL is defined within the scope of the current RVC framework. This RVC-BSDL also includes possibilities for further extensions, which are necessary to provide complete description of video bitstreams. RVC-BSDL specified normatively within the scope of the RVC framework is provided in ISO/IEC 23001-4.

The decoder configuration specified using FNL, together with the specification of the bitstream syntax using RVC-BSDL fully specifies the ADM and provides an "executable" model of the RVC decoder description.

The instantiated ADM includes the information about the selected FUs and how they should be connected. As already mentioned, the FND with the network connection information is expressed by using FNL. Furthermore, the RVC framework specifies and uses a dataflow-oriented language called RVC-CAL for describing FUs' behavior. The normative specification of RVC-CAL is provided in ISO/IEC 23001-4. The ADM is the behavioral model that should be referred to in order to implement any RVC conformant decoder. Any RVC compliant decoding solution/implementation can be achieved by using proprietary non-normative tools and mechanisms that yield decoders that behave equivalent to the RVC ADM.

The decoder description, the MPEG tool library, and the associated instantiation of an ADM are normative. More precisely, the ADM is intended to be normative in terms of a behavioral model. In other words what is normative is the input/output behavior of the complete ADM as well as the input/output behavior of all the FUs that are included in the ADM.

Information technology — MPEG video technologies —

Part 4:
Video tool library

1 Scope

This part of ISO/IEC 23002 defines the description of the MPEG video tool library (VTL) based on the decoder description specified in ISO/IEC 23001-4. This tool library defines the specification of FUs, which are sufficient to build complete decoding solutions according to the following coding standards:

— ISO/IEC 14496-2 (MPEG-4 Simple Profile), and

— ISO/IEC 14496-10 (MPEG-4 AVC Constrained Baseline Profile).

The objective of ISO/IEC 23001-4 is to define the general framework principles, and this part of ISO/IEC 23002 defines the MPEG VTL that includes relevant tools (or FUs) from the existing MPEG coding standards. Each FU is defined in the form of a textual description, which can be found in 4.1. The input and output behavior follows the conventions described in Clause 5 (general-purpose FUs), Clause 6 (MPEG-4 FUs), and Clause 7 (MPEG-4 AVC FUs).

This part of ISO/IEC 23002 compliant implementations can be designed using any software or hardware language and components. The reference software for the textual specification of FUs is written in RVC-CAL language of which a formal syntax is provided in ISO/IEC 23001-4, and which will be defined in Amendment 1 to ISO/IEC 23002-4.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14496-2:2004, *Information technology — Coding of audio-visual objects — Part 2: Visual*

ISO/IEC 23001-4, *Information technology — MPEG systems technologies — Part 4: Codec configuration representation*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 23001-4 apply.

4 FU description convention

4.1 FU interfaces

As shown in Table 1, each FU is described with the following elements;

- **FU Name**: Name to represent the functional unit in this specification. The name of the FU is normative and follows the naming convention described in Annex A.
- **Description**: Textual explanation to describe the functionality of the FU. The description must be concise. The precise normative behaviour of the algorithm (input/output, timing etc.) is specified by the the RVC-CAL reference code in Amendment 1.
- **Profiles@levels supported**: The profiles@level supported for this functional unit. It may append that a given range of values makes the FU behave for a given profile@level and another range of values makes the FU behave for another profile@level.
- **Input**: A token that is entering the FU through the designated input port. The token type refers to the token pool described in 4.3. The 'name' field indicates the input port.
- **Output**: A token that is coming out of the FU through the designated output port. The 'name' field indicates the output port.
- **Parameter (*optional*)**: Parameters are optionally described to adjust the behavior of the FU. All the parameters must be specified with name, description and range.

Table 1 — Template of description of an FU (example)

FU Name	e.g. Algo_IDCT2D_ISOIEC_23002_1	
Description	e.g.This module computes the 8x8 Inverse Discrete Cosine Transform (IDCT) defined as $f(x,y)=\frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{2N}\cos\frac{(2y+1)v\pi}{2N}$ with u, v, x, y = 0, 1, 2, …, N-1 where x, y are spatial coordinates in the sample domain u, v are coordinates in the transform domain $C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u,v=0 \\ 1 & \text{otherwise} \end{cases}$ It inputs a list of 64 coefficients and outputs a list of 64 decoded coefficients.	
Profiles@levels supported	e.g. MPEG-4 SP	
Input		
Name	**Token**	
e.g. X	e.g. BLOCK token	
Output		
Name	**Token**	
e.g. Y	e.g. BLOCK token	
Parameter		
Name	**Description**	**Range**

4.2 FU IDs

FU of the specific functionality is identified by its unique identification number. Table 2 lists IDs and names of all FUs in VTL. IDs and names are used in FND to select FUs.

Table 2 — List of FUs and their IDs

ID	FU Name
1	Algo_SynP_Generic
2	Algo_MVR_MedianOfThreeLeftAndTopAndTopRight
3	Algo_MVSequence_LeftAndTopAndTopRight
4	Mgnt_Splitter_420_TYPE
5	Algo_VLDtableB6_MPEG4Part2
6	Algo_VLDtableB7_MPEG4Part2
7	Algo_VLDtableB8_MPEG4Part2
8	Algo_VLDtableB12_MPEG4Part2
9	Algo_VLDtableB13_MPEG4Part2
10	Algo_VLDtableB14_MPEG4Part2
11	Algo_VLDtableB15_MPEG4Part2
12	Algo_VLDtableB16_MPEG4Part2
13	Algo_VLDtableB17_MPEG4Part2
14	Algo_IQ_QSAndQmatrixMp4vOrH263Scaler
15	Algo_DCRAddr_ThreeLeftTop_8x8
16	Algo_DCRAddr_ThreeLeftTop_16x16
17	Algo_DCRInvPred_CHROMA_8x8
18	Algo_DCRInvPred_LUMA_16x16
19	Algo_IS_ZigzagOrAlternateHorizontalVertical_8x8
20	Algo_IAP_AdaptiveHorizontalOrVerticalPred_8x8
21	Algo_IAP_AdaptiveHorizontalOrVerticalPred_16x16
22	Algo_IDCT2D_ISOIEC_23002_1
23	Mgnt_DCSplit
24	Mgnt_FBMgnt_FBAddr
25	Algo_PictureReconstruction_Saturation
26	Algo_Interp_HalfpelBilinearRoundingControl
27	Algo_NALU FU
28	Algo_Synp_AVC FU
29	Algo_BlockExpand_AVC FU
30	Algo_BlockSplit_AVC FU
31	Algo_IntraPred_Split FU
32	Algo_IS_Zigzag_4x4 FU
33	Algo_DCR_Hadamard_LUMA_IHT1d FU
34	Algo_Transpose4x4 FU
35	Algo_DCR_Hadamard_LUMA_Reordering FU
36	Algo_DCR_Hadamard_LUMA_Scaling FU
37	Algo_DCR_Hadamard_CHROMA FU
38	Algo_IT4x4_1d FU
39	Algo_IT4x4_Addshift FU
40	Algo_IntraPred_LUMA_16x16 FU
41	Algo_IntraPred_LUMA_4x4 FU
42	Algo_Merge_4x4_to_16x16 FU
43	Algo_IQ_QSAndSLAndIDCTScaler_4x4 FU
44	Mgnt_IQ_INTRA16x16 FU
45	Mgnt_Select_3

46	Algo_Merge_4x4_to_8x8 FU
47	Algo_IntraPred_Add FU
48	Algo_IntraPred_CHROMA FU
49	Mgnt_IntraMgnt_Intra4x4
50	Mgnt_IQ_Chroma FU
51	Mgnt_DBF FU
52	Algo_DBF_AdaptiveFilter_AVC FU
53	Algo_Interp_EighthPelBilinear FU
54	Algo_Interp_SeparableSixTapQuarterPelAVC FU
55	Algo_Interp_split_MB FU
56	Algo_Interp_split_MB_C FU
57	Algo_MVR_MultiFrameAdaptive FU
58	Mgnt_DPB_without_adaptiveFilter FU
59	Mgnt_Buffer_Neighbor_FullMb FU
60	Mgnt_Buffer_Neighbor_4x4 FU
61	Algo_MMCO
62	Mgnt_FBAddr_Chroma_MxN FU
63	Mgnt_Interp_FBAddr_Luma_MxN FU
64	Mgnt_POC FU
65	Mgnt_MVR FU
66	Algo_Add FU

4.3 Token pool

Every token is listed in the 'token pool' that is the table of managing all tokens used in VTL. To facilitate the feasibility of connections among input and output ports of different FUs described in this specification, Table 3 lists all data elements (called "token", which is used throughout this document). The ID field here is informative and used for easy lookup.

Table 3 — List of all token types that are used in the descriptions of FUs in this section.

ID & Name	Description
1 BIT	Token which value is 0 or 1. The bits belongs to the non-decoded bitstream
2 ACKNOWLEDGMENT	Boolean token (True or False) indicating an acknowledgment. True means it is OK. False, it is not OK.
3 MCBPC	Token representing the MCBPC element of syntax
4 CBPY	Token representing the CBPY element of syntax
5 DCT_DC_SIZE	Token representing the element of syntax DCT_DC_SIZE
6 DCT_DC_DIFF	Token representing the element of syntax DCT_DC_DIFF
7 RUN	Token representing the RUN value in the decoding of the DCT coefficients
8 VALUE	Token representing the VALUE value in the decoding of the DCT coefficients
9 LAST	Token representing the LAST value in the decoding of the DCT coefficients
10 MEM_ADDRESS	Token representing an address in the memory of the frames
11 MEM_DATA	Token representing a data stored in the memory of the frames

12 WIDTH	Token representing the width value of video frame in pixels
13 HEIGHT	Token representing the height value of video frame in pixels
14 SIZE	Token representing the size of the current frame in macroblock
15 DC	Tokens representing the DC coefficients. Each token represent one coefficient
16 AC	Token representing AC coefficients without DC coefficients
17 BLOCK	Token representing BLOCK that consists of 8x8 pixels
18 MB	Token representing a macroblock that consists of BLOCKs
19 MVD	Tokens representing the motion vector differences decoded by the syntax parsing process
20 MV	Tokens representing the coordinates of the motion vectors
21 QUANT	Token representing the QUANT value of quantization
22 COORDINATE	Token representing coordinates of block or macroblocks
23 DISPLACEMENT	Token representing the displacement between pixels (e.g. half- or quarter-pixel)
24 SIGN	Token representing a sign.
25 ROUND	Boolean token (True or False) indicating whether rounding is to be made or not
26 INTRA_MODE	Boolean token (True or False) indicating INTRA or INTER
27 ACCODED	Boolean token (True or False) indicating whether AC is coded or not
28 ACPRED	Boolean token (True or False) indicating whether AC prediction is made or not
29 ACPRED_DIR	Token representing the order of prediction of the AC coefficients
30 MOTION	Boolean token (True or False) indicating whether motion predication is made or not
31 FOURMV	Boolean token (True or False) indicating whether FOURMV is to be used or not
32 F_CODE	Token representing a value of FCODE of VOP to specify the range of motion vectors
33 RBSP	Token representing the data in the Raw Byte Sequence Payload
34 NAL_SIZE	Token representing the size in byte of a Network Abstraction Layer unit
35 PART_ID	Token representing the identifier for a partition of a macroblock
36 PART_WIDTH	Token representing the width in pixel for a partition of a macroblock
37 PART_HEIGHT	Token representing the height in pixel for a partition of a macroblock
38 PART_SIZE	Token representing the size in pixel for a partition of a macroblock, first the width of the partition, then the height.
39 REF_ID	Token representing the identification of the decoded reference frame in memory

40 MB_ID	Token representing the number that identifies a macroblock in a frame. The macroblocks are counted in a frame using raster scan order.
41 POC	Token representing the index of the frame to display
42 REF_ORDER	Token representing the index of frames to store into long frame reference and short term reference
43 MMCO	Token representing the order of the index for frame to store in memory
44 PRED_MODE_INTRA	Token representing the prediction mode of an intra macroblock
45 MB_TYPE	Token representing the type of prediction used by a macroblock (Intra, Intra 4x4 or Inter)
46 FRACTION	Token representing the MV offset in quarter-pel unit
47 ALPHA_OFFSET	Token representing the offset used in accessing the α and tC0 deblocking filter tables for filtering operations
48 BETA_OFFSET	Token representing the offset used in accessing the β deblocking filter table for filtering operations
49 CBP_BLK	Token representing which of the sixteen 4x4 luma blocks of a macroblock may contain non-zero transform coefficient levels
50 SCALE	Token representing scaling value for quantization
51 DB_SAMPLE	Token representing sample for deblocking filter
52 BS	Token representing boundary strength for deblocking filter

5 General-purpose FUs

5.1 Syntax parsing

5.1.1 Generic syntax parser

FU Name	Algo_SynP_Generic	
Description	This is a generic syntax parser that needs BSD as an input. Input and output port will be defined as the information in the BSD.	
Profiles@levels supported		
Input		
Name	**Token**	
Output		
Name	**Token**	
Parameter		
Name	**Description**	**Range**

6 FUs for MPEG-4 Simple Profile

6.1 Syntax parsing

6.1.1 Algo_MVR_MedianOfThreeLeftAndTopAndTopRight

FU Name	Algo_MVR_MedianOfThreeLeftAndTopAndTopRight	
Description	This module computes the motion vectors from the motion vector differences and the type of encoding of the 8x8 block. The prediction of the motion vector is based on the median value of the motion of three previously decoded blocks (the left, top and top right blocks). The FOURMV, F_CODE, MOTION, VOPMODE and WIDTH indicate how the current 8x8 block is coded. The A tokens are indices indicating the coordinates of the blocks (top, left, top-right) used for the prediction. This FU inputs the motion vectors differences output by the parser and generates the value of the motion vectors (MV output) for each 8x8 block. For each block, the X coordinate followed by the Y coordinates are generated.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
A	COORDINATE token	
FOURMV	FOURMV token	
F_CODE	F_CODE token	
MOTION	MOTION token	
MVIN	MVD token	
VOPMODE	INTRA_MODE token	
WIDTH	WIDTH token	
Output		
Name	**Token**	
MV	MV token	
Parameter		
Name	**Description**	**Range**
MAXW_IN_MB	Maximum width of the frame in macroblock	
MB_COORD_SZ	Size in bits of some variables	[0..32]
MV_SZ	Size in bits of port MV	[0..32]
VOP_FCODE_FOR_LENGTH	Size in bits of F_CODE	3
VOL_WIDTH_LENGTH	Size in bits of WIDTH	

6.1.2 Algo_MVSequence_LeftAndTopAndTopRight

FU Name	Algo_MVSequence_LeftAndTopAndTopRight	
Description	This module computes the sequence of coordinates of the different blocks necessary for the prediction of the motion vectors. From the type of encoding of the block (given by the FOURMV, MOTION, VOPMODE, and WIDTH), the FU generates the coordinates of the blocks (on the A port) which will be used by the FU charged of reconstructing the motion vectors for each 8x8 block.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
FOURMV	FOURMV token	
MOTION	MOTION token	
VOPMODE	INTRA_MODE token	
WIDTH	WIDTH token	
Output		
Name	**Token**	
A	COORDINATE token	
Parameter		
Name	**Description**	**Range**
MAXW_IN_MB	Maximum width of the frame in macroblock	
MB_COORD_SZ	Size in bits of some variables	[0..32]
VOL_WIDTH_LENGTH	Size in bits of WIDTH	

6.1.3 Mgnt_Splitter_420_TYPE

FU Name	Mgnt_Splitter_420_TYPE	
Description	This module distributes each VOPMODE, ACCODED, ACPRED, and MOTION for each Y, U, and V components sequentially.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
VOPMODE	INTRA_MODE token	
MOTION	MOTION token	
ACCODED	ACCODED token	
ACPRED	ACPRED token	
Output		
Name	**Token**	
VOPMODE	INTRA_MODE token	
MOTION	MOTION token	
ACCODED	ACCODED token	
ACPRED	ACPRED token	
Parameter		
Name	**Description**	**Range**

6.1.4 Algo_VLDtableB6_MPEG4Part2

FU Name	Algo_VLDtableB6_MPEG4Part2	
Description	This Functional Unit decodes the mcbpc element of syntax of a MPEG-4 conformant bitstream. It applies in the case of intra mode. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.6.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	MCBPC token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.5 Algo_VLDtableB7_MPEG4Part2

FU Name	Algo_VLDtableB7_MPEG4Part2	
Description	This Functional Unit decodes the mcbpc element of syntax of a MPEG-4 conformant bitstream. It applies in the case of inter mode. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.7.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	MCBPC token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.6 Algo_VLDtableB8_MPEG4Part2

FU Name	Algo_VLDtableB8_MPEG4Part2	
Description	This Functional Unit decodes the cbpy element of syntax of a MPEG-4 conformant bitstream. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.8.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	CBPY token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.7 Algo_VLDtableB12_MPEG4Part2

FU Name	Algo_VLDtableB12_MPEG4Part2	
Description	This Functional Unit decodes the elements of syntax relative to the motion vectors in a MPEG-4 conformant bitstream ("horizontal_mv_data" and "vertical_mv_data"). It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.12.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	MVD token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.8 Algo_VLDtableB13_MPEG4Part2

FU Name	Algo_VLDtableB13_MPEG4Part2	
Description	This Functional Unit decodes the “dct_dc_size” element of syntax in a MPEG-4 conformant bitstream. The decoding applies only for the luminance macroblocks. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.13.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	DCT_DC_SIZE token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.9 Algo_VLDtableB14_MPEG4Part2

FU Name	Algo_VLDtableB14_MPEG4Part2	
Description	This Functional Unit decodes the “dct_dc_size” element of syntax in a MPEG-4 conformant bitstream. The decoding applies only for the chrominance macroblocks. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.14.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	DCT_DC_SIZE token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.10 Algo_VLDtableB15_MPEG4Part2

FU Name	Algo_VLDtableB15_MPEG4Part2	
Description	This Functional Unit decodes the “dct_dc_differencial” element of syntax in a MPEG-4 conformant bitstream. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.15.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
DATA	DCT_DC_DIFF token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]

6.1.11 Algo_VLDtableB16_MPEG4Part2

FU Name	Algo_VLDtableB16_MPEG4Part2	
Description	This Functional Unit decodes the INTRA coefficients as elements of syntax in a MPEG-4 conformant bitstream. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.16.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
RUN	RUN token	
VALUE	VALUE token	
LAST	LAST token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]
SAMPLE_SZ	Size in bits of the data used inside	[0..32]
SAMPLE_COUNT_SZ	Size in bits of the data variable used inside	[0..32]

6.1.12 Algo_VLDtableB17_MPEG4Part2

FU Name	Algo_VLDtableB17_MPEG4Part2	
Description	This Functional Unit decodes the INTER coefficients as elements of syntax in a MPEG-4 conformant bitstream. It decodes the bits as specified in ISO/IEC 14496-2:2004, Table B.17.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
BITS	BIT token	
Output		
Name	**Token**	
FINISH	ACKNOWLEDGMENT token	
RUN	RUN token	
VALUE	VALUE token	
LAST	LAST token	
Parameter		
Name	**Description**	**Range**
VLD_DATA_SZ	Size in bits of the data used inside	[0..32]
VLD_ADDR_SZ	Size in bits of the address variable used inside	[0..32]
SAMPLE_SZ	Size in bits of the data used inside	[0..32]
SAMPLE_COUNT_SZ	Size in bits of the data variable used inside	[0..32]

6.2 Texture decoding

6.2.1 Algo_IQ_QSAndQmatrixMp4vOrH263Scaler

FU Name	Algo_IQ_QSAndQmatrixMp4vOrH263Scaler	
Description	This module computes inverse quantization of AC for 8x8 blocks. Supports both MPEG and H.263 Inverse Quantization modes.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
DC	DC token	
AC	AC token	
QP	QUANT token	
Output		
Name	**Token**	
OUT	BLOCK token	
Parameter		
Name	**Description**	**Range**
QUANT_SZ	Size in bits of the QP port	[0..32]
SAMPLE_SZ	Size in bits of the DC, AC and OUT ports	[0..32]

6.2.2 Algo_DCRAddr_ThreeLeftTop_8x8

FU Name	Algo_DCRAddr_ThreeLeftTop_8x8	
Description	This module calculates the addresses of the three neighboring blocks for the current 8x8 block used for DC prediction. If any of the neighbor is not coded for some reason (either outside frame boundaries or skipped by encoder), the address is set as zero. Otherwise the lowest two bits specify the 8x8 component in the macroblock and the other higher order bits specify the macroblock index using circular buffer addressing.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
VOPMODE	INTRA_MODE token	
WIDTH	WIDTH token	
Output		
Name	**Token**	
A	COORDINATE token	
B	COORDINATE token	
C	COORDINATE token	
Parameter		
Name	**Description**	**Range**
MB_COORD_SZ	Size in bits of the token ADDR	[0..32]
MAXW_IN_MB	Maximum width of the frame in macroblock	
VOL_WIDTH_LENGTH	Size in bits of WIDTH	

6.2.3 Algo_DCRAddr_ThreeLeftTop_16x16

<table>
<tr><td><b>FU Name</b></td><td>Algo_DCRAddr_ThreeLeftTop_16x16</td></tr>
<tr><td><b>Description</b></td><td>This module calculates the addresses of the three neighboring blocks for the current 8x8 block used for DC prediction. This module manages groups of four 8x8 blocks as follows:
+---+---+
| 0 | 1 |
+---+---+
| 2 | 3 |
+---+---+
If any of the neighbor is not coded for some reason (either outside frame boundaries or skipped by encoder), the address is set as zero. Otherwise the lowest two bits specify the 8x8 component in the macroblock and the other higher order bits specify the macroblock index using circular buffer addressing.</td></tr>
<tr><td><b>Profiles@levels supported</b></td><td>MPEG-4 SP</td></tr>
<tr><td colspan="2"><b>Input</b></td></tr>
<tr><td><b>Name</b></td><td><b>Token</b></td></tr>
<tr><td>VOPMODE</td><td>INTRA_MODE token</td></tr>
<tr><td>WIDTH</td><td>WIDTH token</td></tr>
</table>

Output		
Name	**Token**	
A	COORDINATE token	
B	COORDINATE token	
C	COORDINATE token	
Parameter		
Name	**Description**	**Range**
MB_COORD_SZ	Size in bits of the token ADDR	[0..32]
MAXW_IN_MB	Maximum width of the frame in macroblock	
VOL_WIDTH_LENGTH	Size in bits of WIDTH	

6.2.4 Algo_DCRInvPred_CHROMA_8x8

FU Name	Algo_DCRInvPred_CHROMA_8x8	
Description	This module reconstructs the DC coefficient of the current 8x8 block based on the gradients between neighboring block DC coefficients (see ISO/IEC 14496-2). This module also forwards the decoded prediction direction to the inverse AC prediction (IAP) module and a pointer to the neighboring block used for the prediction. Since inverse quantization is necessary to reconstruct the DC coefficient, the decoded quantization parameter is forwarded to the inverse quantization module. This FU applies for 8x8 blocks of chrominance.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
A	COORDINATE token	
B	COORDINATE token	
C	COORDINATE token	
ACCODED	ACCODED token	
ACPRED	ACPRED token	
QFS_DC	DC token	
QUANT	QUANT token	
VOPMODE	INTRA_MODE token	
Output		
Name	**Token**	
QF_DC	DC token	
PTR	COORDINATE token	
AC_PRED_DIR	ACPRED_DIR token	
SIGNED	SIGN token	
Parameter		
Name	**Description**	**Range**
DCVAL	Size in bits of the QFS_DC and QF_DC ports	
MAXW_IN_MB	Maximum width of the frame in macroblock	
SAMPLE_SZ	Size in bits of the A, B, C, and PTR ports	[0..32]
QUANT_SZ	Size in bits of QUANT	[0..32]

6.2.5 Algo_DCRInvPred_LUMA_16x16

FU Name	Algo_DCRInvPred_LUMA_16x16	
Description	This module reconstructs the DC coefficient of the current 8x8 block based on the gradients between neighboring block DC coefficients (see ISO/IEC 14496-2). This module also forwards the decoded prediction direction to the inverse AC prediction (IAP) module and a pointer to the neighboring block used for the prediction. Since inverse quantization is necessary to reconstruct the DC coefficient, the decoded quantization parameter is forwarded to the inverse quantization module. This FU applies for four 8x8 blocks of luminance.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
A	COORDINATE token	
B	COORDINATE token	
C	COORDINATE token	
ACCODED	ACCODED token	
ACPRED	ACPRED token	
QFS_DC	DC token	
QUANT	QUANT token	
VOPMODE	INTRA_MODE token	
Output		
Name	**Token**	
QF_DC	DC token	
PTR	COORDINATE token	
AC_PRED_DIR	ACPRED_DIR token	
SIGNED	SIGN token	
Parameter		
Name	**Description**	**Range**
DCVAL	Size in bits of the QFS_DC and QF_DC ports	
MAXW_IN_MB	Maximum width of the frame in macroblock	
SAMPLE_SZ	Size in bits of the A, B, C, and PTR ports	[0..32]
QUANT_SZ	Size in bits of QUANT	[0..32]

6.2.6 Algo_IS_ZigzagOrAlternateHorizontalVertical_8x8

FU Name	Algo_IS_ZigzagOrAlternateHorizontalVertical_8x8
Description	This module inverts the one-dimensional array of coefficients ordered in zigzag (AC_PRED_DIR=0), alternate vertical (AC_PRED_DIR=1) or alternate horizontal (AC_PRED_DIR=2) scan to 2D raster order. It inputs a list of 64 integer coefficients (one per 8x8 block) and outputs the ordered list of integer according to the value of the token AC_PRED_DIR.
Profiles@levels supported	MPEG-4 SP

Input		
Name	**Token**	
AC_PRED_DIR	ACPRED_DIR token	
QFS_AC	AC token	
Output		
Name	**Token**	
PQF_AC	AC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the QFS_AC and PQF_AC ports	[0..32]

6.2.7 Algo_IAP_AdaptiveHorizontalOrVerticalPred_8x8

FU Name	Algo_IAP_AdaptiveHorizontalOrVerticalPred_8x8	
Description	This module computes inverse AC prediction for specific AC coefficients of 8x8 blocks that have been flagged in the bitstream as coded in this fashion. It inputs a list of 63 AC coefficients received in a horizontal raster manner after being re-ordered by the inverse scan block and the addresses of the 8x8 block used for the prediction in the encoder. It outputs a list of 63 reconstructed AC coefficients sent in a horizontal raster man. The AC_PRED_DIR token communicates the direction of prediction: • AC_PRED_DIR = -2: NEWVOP flag • AC_PRED_DIR = -1: An uncoded block so skip inverse AC prediction • AC_PRED_DIR = 0: No inverse AC prediction but use zigzag inverse scan • AC_PRED_DIR = 1: Prediction form the left and use alternate vertical scan • AC_PRED_DIR = 2: Prediction form the top and use alternate horizontal scan	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
PQF_AC	AC token	
PTR	COORDINATE token	
AC_PRED_DIR	ACPRED_DIR token	
Output		
Name	**Token**	
QF_AC	AC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the QF_AC port	[0..32]
MAXW_IN_MB	Maximum width of the frame in MB	
MB_COORD_SZ	Size in bits of the variables PTR port	[0..32]

6.2.8 Algo_IAP_AdaptiveHorizontalOrVerticalPred_16x16

FU Name	Algo_IAP_AdaptiveHorizontalOrVerticalPred_16x16	
Description	This module computes inverse AC prediction for specific AC coefficients of a 16x16 blocks that have been flagged in the bitstream as coded in this fashion. It inputs four lists of 63 AC coefficients received in a horizontal raster manner after being re-ordered by the inverse scan block and the addresses of the four 8x8 block used for the prediction in the encoder. It outputs four lists of 63 reconstructed AC coefficients sent in a horizontal raster man. The AC_PRED_DIR token communicates the direction of prediction: • AC_PRED_DIR = -2 => NEWVOP flag • AC_PRED_DIR = -1 => An un-coded block so skip inverse AC prediction • AC_PRED_DIR = 0 => No inverse AC prediction but use zigzag inverse scan • AC_PRED_DIR = 1 => Prediction form the left and use alternate vertical scan • AC_PRED_DIR = 2 => Prediction form the top and use alternate horizontal scan	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
PQF_AC	AC token	
PTR	COORDINATE token	
AC_PRED_DIR	ACPRED_DIR token	
Output		
Name	**Token**	
QF_AC	AC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the QF_AC port	[0..32]
MAXW_IN_MB	Maximum width of the frame in MB	
MB_COORD_SZ	Size in bits of the variables PTR port	[0..32]

6.2.9 Algo_IDCT2D_ISOIEC_23002_1

FU Name	Algo_IDCT2D_ISOIEC_23002_1
Description	This module computes the 8x8 Inverse Discrete Cosine Transform (IDCT) defined as $$f(x,y)=\frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{2N}\cos\frac{(2y+1)v\pi}{2N}$$ with u, v, x, y = 0, 1, 2, …, N-1 where x, y are spatial coordinates in the sample domain u, v are coordinates in the transform domain $$C(u),C(v)=\begin{cases}\frac{1}{\sqrt{2}} & \text{for } u,v=0\\ 1 & \text{otherwise}\end{cases}$$ It inputs a list of 64 coefficients and outputs a list of 64 decoded coefficients.

Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
X	BLOCK token	
Output		
Name	**Token**	
Y	BLOCK token	
Parameter		
Name	**Description**	**Range**

6.2.10 Mgnt_DCSplit

FU Name	Mgnt_DCSplit	
Description	This module separates the DC coefficient from the AC coefficients. It takes as an input a list of 64 tokens and outputs on the DC port the DC coefficient (the first one) and the AC coefficient (the other 63) on the AC port.	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
IN	BLOCK token	
Output		
Name	**Token**	
AC	AC token	
DC	DC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the AC, DC and IN ports	[0..32]

6.3 Motion compensation

6.3.1 Mgnt_FB

FU Name	Mgnt_FB
Description	This module is a frame buffer. It saves data WD at address WA. It outputs the data RD located at address RA
Profiles@levels supported	MPEG-4 SP
Input	
Name	**Token**
WA	MEM_ADDRESS token
WD	MB token
RA	MEM_ADDRESS token
Output	
Name	**Token**
RD	MB token

Parameter		
Name	Description	Range
BUF_SZ	Size in bits of the AC, DC and IN ports	[0..32]
ADDR_SZ	Size in bits of the addresses ports (RA and WA)	[0..32]
PIX_SZ	Size in bits of the data ports (WD and RD)	[0..32]

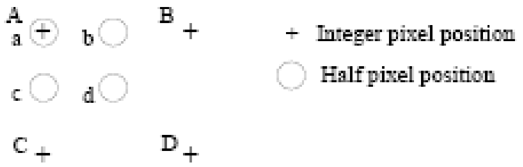
6.3.2 Mgnt_FBAddr

FU Name	Mgnt_FBAddr	
Description	This module generates addresses for the frame buffer block. Write addresses are used to save the current pixels for retrieval and read addresses are used to retrieve interpolation pixel values. This FU supports the following layouts: - if LAYOUT = 0, it supports 8x8 blocks - If LAYOUT = 1, it supports 16x16 blocks	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
HEIGHT	HEIGHT token	
MOTION	MOTION token	
MV	MV token	
ROUND	ROUND token	
VOPMODE	INTRA_MODE token	
WIDTH	WIDTH token	
Output		
Name	**Token**	
RA	MEM_ADDRESS token	
WA	MEM_ADDRESS token	
HALFPEL	DISPLACEMENT token	
Parameter		
Name	**Description**	**Range**
SEARCHWIN_IN_MB	Size of the search window in macroblock	
MAXW_IN_MB	Width of the frame in macroblock	
MAXH_IN_MB	Height of the frame in macroblock	
ADDR_SZ	Number of bits used to represent the "*address*" data type	[0..32]
FLAG_SZ	Number of bits used to represent the "*flag*" data type	[0..32]
MV_SZ	Number of bits used to represent the "*motion vectors*" data type	[0..32]
MB_COORD_SZ	Number of bits used to represent the "*macroblock coordinates*" data type	[0..32]
LAYOUT	Number indicating a layout (see Description)	
VOL_WIDTH_LENGTH	Size in bits of WIDTH	
VOL_HEIGHT_LENGTH	Size in bits of HEIGHT	

6.3.3 Algo_PictureReconstruction_Saturation

FU Name	Algo_PictureReconstruction_Saturation	
Description	This module adds texture pixels (TEX) and MC prediction pixels (MOT) in order to output the decoded pixels. For each ACCODED and WIDTH token inputted and according to the type of encoding of the block under consideration, the FU consumes either: - one token from the MOT input if the block is only a "motion" block - one token from the TEX input if the block is only "texture" block - one token from TEX input and one token from MOT input otherwise	
Profiles@levels supported	MPEG-4 SP	
Input		
Name	**Token**	
MOT	MB token	
TEX	MB token	
ACCODED	ACCODED token	
VOPMODE	INTRA_MODE token	
Output		
Name	**Token**	
VID	MB token	
Parameter		
Name	**Description**	**Range**
PIX_SZ	Size in bits of the pixels data	[0..32]

6.3.4 Algo_Interp_HalfpelBilinearRoundingControl

FU Name	Algo_Interp_HalfpelBilinearRoundingControl
Description	This module interpolates the pixels in case of a displacement between two frames of half a pixel. This is done according to the following scheme:  A a b B c d C D Integer pixel position Half pixel position a = A, b = (A + B + 1 - rounding_control) / 2 c = (A + C + 1 - rounding_control) / 2, d = (A + B + C + D + 2 - rounding_control) / 4
Profiles@levels supported	MPEG-4 SP
Input	
Name	**Token**
RD	MB token
halfpel	DISPLACEMENT token
Output	
Name	**Token**
MOT	MB token

Parameter		
Name	**Description**	**Range**
FLAG_SZ	Number of bits used to represent the "*flag*" data type	[0..32]
PIX_SZ	Number of bits used to represent the "*pixels*" data type	[0..32]

7 FUs for MPEG-4 AVC Constrained Baseline Profile

7.1 Syntax parsing

7.1.1 Algo_NALU FU

FU Name	Algo_NALU	
Description	This module removes emulation_prevention_three_byte (0x03) and sends to its output RBSP bytes and the number of rbsp bytes between 2 NALs.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
bits8	BIT token	
Output		
Name	**Token**	
Bits_rbsp8	RBSP token	
Nb_rbsp_byte	NAL_SIZE token	
Parameter		
Name	**Description**	**Range**

7.1.2 Algo_Synp_AVC FU

FU Name	Algo_Synp_AVC
Description	This module analyzes a sequence of tokens and realizes Parsing for MPEG-4 AVC syntax
Profiles@levels supported	MPEG-4 AVC Constrained BP
Input	
Name	**Token**
Bits_rbsp8	RBSP token
Nb_rbsp_byte	NAL_SIZE token
Output	
Name	**Token**
Cbp_blk	CBP_BLK token
CurrMbAddr	MB_ID token
pix_I_PCM	BIT token
IntraPredMode	BIT token
IntraPredModeC	BIT token
LAST	LAST token

LFDisable	BIT token	
MMCO	MMCO token	
MbIntraFlag	BIT token	
MbPartHeigth	PART_HEIGHT token	
MbPartIdx	PART_ID token	
MbPartWidth	PART_WIDTH token	
MvRes	MVD token	
POC	POC token	
PicSizeInMb	SIZE token	
PicWidthInMb	WIDTH token	
QP	QUANT token	
QP_Cb	QUANT token	
QP_Cr	QUANT token	
RUN	RUN token	
RefIdx	REF_ID token	
RefReordering	BIT token	
ScalingList	SCALE token	
VALUE	VALUE token	
Alpha_offset	ALPHA_OFFSET token	
Beta_offset	BETA_OFFSET token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the VALUE and Block ports	[0..16]

7.1.3 Algo_BlockExpand_AVC FU

FU Name	Algo_BlockExpand_AVC	
Description	This module decodes the DCT coefficients as elements of syntax in a MPEG-4 AVC Constrained BP conformant bitstream.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
LAST	LAST token	
RUN	RUN token	
VALUE	VALUE token	
Output		
Name	**Token**	
Block	BLOCK token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the VALUE and Block ports	[0..16]

7.1.4 Algo_BlockSplit_AVC FU

FU Name	Algo_BlockSplit_AVC	
Description	This module splits DCT coefficient from LUMA/CHROMA into the DC and AC CHROMA/LUMA coefficients.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
Block	BLOCK token	
Output		
Name	**Token**	
Block_U_AC	AC token	
Block_U_DC	DC token	
Block_V_AC	AC token	
Block_V_DC	DC token	
Block_Y_AC	AC token	
Block_Y_DC	DC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the Block, AC and DC coefficient ports	[0..16]

7.1.5 Algo_IntraPred_Split FU

FU Name	Algo_IntraPred_Split	
Description	This module sends information token of the prediction type (Mb_Type for LUMA and Mb_TypeC for CHROMA) and mode (PredMode) used by the current macroblock according to MbIntraFlag and IntraPredMode Input.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
IntraPredMode	PRED_MODE_INTRA token	
CurrMbAddr	MB_ID token	
MbIntraFlag	BIT token	
PicWidthInMb	WIDTH token	
Output		
Name	**Token**	
Mb_Type	MB_TYPE token	
Mb_TypeC	MB_TYPE token	
PredMode	PRED_MODE_INTRA token	
Parameter		
Name	**Description**	**Range**

7.2 Texture decoding

7.2.1 Algo_IS_Zigzag_4x4 FU

FU Name	Algo_IS_Zigzag_4x4	
Description	This module inverts the one-dimensional array of coefficients ordered in zigzag scan to 2D raster order. It inputs a list of 16 integer coefficients (one per 4x4 block) and outputs the ordered list of integer values.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
levarr	BLOCK token	
Output		
Name	**Token**	
Lev2d	BLOCK token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the levarr and Lev2d ports	[0…16]

7.2.2 Algo_DCR_Hadamard_LUMA_IHT1d FU

FU Name	Algo_DCR_Hadamard_LUMA_IHT1d	
Description	This module computes 1 dimensional 4x4 Inverse Hadamard Transform for DC luminance coefficients of an intra_16x16 prediction block as a part of 2 dimensional 4x4 Inverse Hadamard Transform.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	DC token	
Output		
Name	**Token**	
Y	DC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X and Y ports	[0…16]

7.2.3 Algo_Transpose4x4 FU

FU Name	Algo_Transpose4x4	
Description	This module transposes 4x4 integer array.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	DC token	
Output		
Name	**Token**	
Y	DC token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X and Y ports	[0…16]

7.2.4 Algo_DCR_Hadamard_LUMA_Reordering FU

<table>
<tr><td><b>FU Name</b></td><td colspan="2">Algo_DCR_Hadamard_LUMA_Reordering</td></tr>
<tr><td><b>Description</b></td><td colspan="2">This module reorders 4x4 DC luminance coefficients of an intra 16x16 prediction macroblock from raster scan order to the order of block number specified in MPEG-4 AVC specification.

The order of block number in a MB
+---+---+---+---+
| 0 | 1 | 4 | 5 |
+---+---+---+---+
| 2 | 3 | 6 | 7 |
+---+---+---+---+
| 8 | 9 |12 |13 |
+---+---+---+---+
|10 |11 |14 |15 |
+---+---+---+---+</td></tr>
<tr><td><b>Profiles@levels supported</b></td><td colspan="2">MPEG-4 AVC Constrained BP</td></tr>
<tr><td colspan="3"><b>Input</b></td></tr>
<tr><td><b>Name</b></td><td colspan="2"><b>Token</b></td></tr>
<tr><td>X</td><td colspan="2">DC token</td></tr>
<tr><td colspan="3"><b>Output</b></td></tr>
<tr><td><b>Name</b></td><td colspan="2"><b>Token</b></td></tr>
<tr><td>Y</td><td colspan="2">DC token</td></tr>
<tr><td colspan="3"><b>Parameter</b></td></tr>
<tr><td><b>Name</b></td><td><b>Description</b></td><td><b>Range</b></td></tr>
<tr><td>SAMPLE_SZ</td><td>Size in bits of the X and Y ports</td><td>[0…16]</td></tr>
</table>

7.2.5 Algo_DCR_Hadamard_LUMA_Scaling FU

<table>
<tr><td><b>FU Name</b></td><td colspan="2">Algo_DCR_Hadamard_LUMA_Scaling</td></tr>
<tr><td><b>Description</b></td><td colspan="2">This module computes inverse quantization of DC luminance coefficients for intra 16x16 prediction macroblock.</td></tr>
<tr><td><b>Profiles@levels supported</b></td><td colspan="2">MPEG-4 AVC Constrained BP</td></tr>
<tr><td colspan="3"><b>Input</b></td></tr>
<tr><td><b>Name</b></td><td colspan="2"><b>Token</b></td></tr>
<tr><td>I</td><td colspan="2">DC token</td></tr>
<tr><td>QP</td><td colspan="2">QUANT token</td></tr>
<tr><td>Scalinglist</td><td colspan="2"><b>SCALE</b> token</td></tr>
<tr><td colspan="3"><b>Output</b></td></tr>
<tr><td><b>Name</b></td><td colspan="2"><b>Token</b></td></tr>
<tr><td>O</td><td colspan="2">DC token</td></tr>
<tr><td colspan="3"><b>Parameter</b></td></tr>
<tr><td><b>Name</b></td><td><b>Description</b></td><td><b>Range</b></td></tr>
<tr><td>QUANT_SZ</td><td>Size in bits of the QP and Scalinglist ports</td><td>[0…8]</td></tr>
<tr><td>SAMPLE_SZ</td><td>Size in bits of the I and O ports</td><td>[0…16]</td></tr>
</table>

7.2.6 Algo_DCR_Hadamard_CHROMA FU

FU Name	Algo_DCR_Hadamard_CHROMA	
Description	This module computes 2 dimensional 2x2 Inverse Hadamard Transform and inverse quantization for DC chrominance coefficients of a macroblock.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
levarr	DC token	
QP	QUANT token	
Scalinglist	SCALE token	
Output		
Name	**Token**	
Cof	DC token	
Parameter		
Name	**Description**	**Range**
QUANT_SZ	Size in bits of the QP and Scalinglist ports	[0…8]
SAMPLE_SZ	Size in bits of the I and O ports	[0…16]

7.2.7 Algo_IT4x4_1d FU

FU Name	Algo_IT4x4_1d	
Description	This module computes 1 dimensional 4x4 Inverse Integer Transform for 4x4 block coefficients according to MPEG-4 AVC specification as a part of 2 dimensional 4x4 Inverse Integer Transform.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	BLOCK token	
Output		
Name	**Token**	
Y	BLOCK token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X and Y ports and internal data	[0…16]

7.2.8 Algo_IT4x4_Addshift FU

FU Name	Algo_IT4x4_Addshift	
Description	This module computes right shifting of input integer value after adding value of 32 as a part of 2 dimensional 4x4 Inverse Integer Transform specified in MPEG-4 AVC specification.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	BLOCK token	
Output		
Name	**Token**	
Y	BLOCK token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X and Y ports and internal data	[0…16]

7.2.9 Algo_IntraPred_LUMA_16x16 FU

FU Name	Algo_IntraPred_LUMA_16x16	
Description	This module computes intra 16x16 predicted block for a 16x16 luminance block according to MPEG-4 AVC intra 16x16 prediction.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
Y_LEFT	COORDINATE token	
Y_UP	COORDINATE token	
Y_UP_LEFT	COORDINATE token	
AVAIL	ACKNOWLEDGMENT token	
PredMode	PRED_MODE_INTRA token	
Output		
Name	**Token**	
MPR	MB token	
Parameter		
Name	**Description**	**Range**
PIX_SZ	Size in bits of the decoded pixel	8

7.2.10 Algo_IntraPred_LUMA_4x4 FU

FU Name	Algo_IntraPred_LUMA_4x4
Description	This module computes intra 4x4 predicted block for a 4x4 luminance block according to MPEG-4 AVC intra 4x4 prediction.
Profiles@levels supported	MPEG-4 AVC Constrained BP
Input	
Name	**Token**
Y_LEFT	COORDINATE token
Y_UP	COORDINATE token

Y_UP_LEFT	COORDINATE token	
AVAIL	ACKNOWLEDGMENT token	
PredMode	PRED_MODE_INTRA token	
Output		
Name	**Token**	
MPR	MB token	
Parameter		
Name	**Description**	**Range**
PIX_SZ	Size in bits of the decoded pixel	8

7.2.11 Algo_Merge_4x4_to_16x16 FU

FU Name	Algo_Merge_4x4_to_16x16	
Description	This module merges 4x4 blocks send in raster scan order into a 16x16 macroblocks.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	MB token	
Output		
Name	**Token**	
Y	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X and Y ports	[0…16]

7.2.12 Algo_IQ_QSAndSLAndIDCTScaler_4x4 FU

FU Name	Algo_IQ_QSAndSLAndIDCTScaler_4x4	
Description	This module computes inverse quantization of 16 luminance (4x4 block) and 4 chrominance (4x4 block) for Cb and Cr components within a macroblock.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
Lev2d	BLOCK token	
QP	QUANT token	
intra_DC_flag	INTRA_MODE token	
ScalingList	SCALE token	
Output		
Name	**Token**	
Cof	BLOCK token	
Parameter		
Name	**Description**	**Range**
QUANT_SZ	Size in bits of the QP and Scalinglist ports	[0…8]
SAMPLE_SZ	Size in bits of the I and O ports	[0…16]
PREDTYPE_SZ	Size in bits of the intra_DC_flag port	[0…2]
NB_4x4	Size in bits of the number of 4x4 blocks in a macroblock for the processing component	[0…4]

7.2.13 Mgnt_IQ_INTRA16x16 FU

FU Name	Mgnt_IQ_INTRA16x16	
Description	This module combines DC coefficients, if available, with AC coefficients within a 16x16 block	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CoefDCR_L	BLOCK token	
Intra_DC_flag	INTRA_MODE token	
CoefAC_L	BLOCK token	
Output		
Name	**Token**	
Lev2d	BLOCK token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the DC, AC and OUT ports	[0…16]
PREDTYPE_SZ	Size in bit of the Intra_DC_flag	[0…2]

7.2.14 Mgnt_Select_3

FU Name	Mgnt_DemuxIntraInter	
Description	This module selects data from one of its input (X0, X1, and X2) and sends it to its output (X) according to Mb_Type.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X0	MB token	
X1	MB token	
X2	MB token	
MbType	MB_TYPE token	
Output		
Name	**Token**	
X	MB token	
Parameter		
Name	**Description**	**Range**
MB_WIDTH	Size in pixel of macroblocks	[8;16]
SAMPLE_SZ	Size in bits of data in X0, X1, X2 and X ports	[0..16]

7.2.15 Algo_Merge_4x4_to_8x8 FU

FU Name	Algo_IntraPred_4x4_to_8x8	
Description	This module merges 4x4 blocks send in raster scan order into an 8x8 macroblocks.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	MB token	
Output		
Name	**Token**	
Y	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X and Y ports	[0...16]

7.2.16 Algo_IntraPred_Add FU

FU Name	Algo_IntraPred_Add	
Description	This module produces to its Z output an addition of prediction macroblock from X with residual macroblock from Y. Negative results are set to 0 and results higher than 255 are set to 255.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	MB token	
Y	MB token	
Output		
Name	**Token**	
Z	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the X, Y and Z ports	8

7.2.17 Algo_IntraPred_CHROMA FU

FU Name	Algo_IntraPred_CHROMA
Description	This module computes an intra chroma predicted block for a chrominance block according to MPEG-4 AVC intra chroma prediction.
Profiles@levels supported	MPEG-4 AVC Constrained BP
Input	
Name	**Token**
C_LEFT	COORDINATE token
C_UP	COORDINATE token
C_UP_LEFT	COORDINATE token
AVAIL	ACKNOWLEDGMENT token
PredMode	PRED_MODE_INTRA token

Output		
Name	**Token**	
MPR	MB token	
Parameter		
Name	**Description**	**Range**
PIX_SZ	Size in bits of the decoded pixel	8

7.2.18 Mgnt_Intra

FU Name	Mgnt_Intra	
Description	This module activates and sends the value of pixels needed for a full macroblock intra prediction to output Y_LEFT, Y_UP, Y_UP_LEFT, the available edge for intra prediction to output AVAIL, the prediction mode to output PredMode_I and the error of the prediction to output Coef_ACR_I. Value of needed pixels are receive from EDGE input.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CurrMbAddr	MB_ID token	
FirstMbInSlice	BIT token	
PicWidthInMb	WIDTH token	
EDGE	MB token	
PredMode	PRED_MODE_INTRA token	
Coef_ACR	MB token	
Output		
Name	**Token**	
AVAIL	BIT token	
Y_LEFT	COORDINATE token	
Y_UP	COORDINATE token	
Y_UP_LEFT	COORDINATE token	
PredMode_I	PRED_MODE_INTRA token	
Coef_ACR_I	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size of RD, C_LEFT, C_UP and C_UP_LEFT ports	[0..16]
MB_WIDTH	Width and height in pixel of macroblock	[8;16]

7.2.19 Mgnt_Intra4x4

FU Name	Mgnt_Intra4x4	
Description	This module activates and sends the value of pixels needed for a 4x4 macroblock intra prediction to output Y_LEFT_4, Y_UP_4, Y_UP_LEFT_4, the available edge for intra prediction to output AVAIL, the prediction mode to output PredMode_4x4 and the error of the prediction to output Coef_ACR_4x4. Value of needed pixels are receive from EDGE for neighboring 16x16 pixels and MB4x4 input for neighboring 4x4 pixels.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CurrMbAddr	MB_ID token	
FirstMbInSlice	BIT token	
PicWidthInMb	WIDTH token	
EDGE	MB token	
PredMode	PRED_MODE_INTRA token	
Coef_ACR	MB token	
MB_4X4	MB token	
Output		
Name	**Token**	
AVAIL	BIT token	
Y_LEFT_4	COORDINATE token	
Y_UP_4	COORDINATE token	
Y_UP_LEFT_4	COORDINATE token	
PredMode_4x4	PRED_MODE_INTRA token	
Coef_ACR_4x4	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size of RD, C_LEFT, C_UP and C_UP_LEFT ports	[0..16]
MB_WIDTH	Width and height in pixel of macroblock	[8;16]

7.2.20 Mgnt_IQ_Chroma FU

FU Name	Mgnt_IQ_Chroma
Description	This module adds the DC coefficient to the AC coefficients
Profiles@levels supported	MPEG-4 AVC Constrained BP
Input	
Name	**Token**
CoefDCR_C	BLOCK token
CoefAC_C	BLOCK token
Output	
Name	**Token**
Lev2d	BLOCK token
intra_DC_flag	INTRA_MODE token

Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the DC, AC and OUT ports	[0..16]
PREDTYPE_SZ	Size in bit of the Intra_DC_flag	[0…2]

7.3 Motion compensation

7.3.1 Mgnt_DBF FU

FU Name	Mgnt_DBF	
Description	This module manages pixels to be deblocked for the deblocking filter FU on the basis of 20x20 or 12x12 blocks for luma and chroma samples, respectively. In addition, it sends the boundary strengths for the deblocking filter FU and eventually outputs the deblocked macroblocks to the decoded picture buffer.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
MB_in	MB token	
DB_O	DB_SAMPLE token	
CurrMbAddr	MB_ID token	
PicWidthInMb	WIDTH token	
MbType	MB_TYPE token	
Cbp_blk	CBP_BLK token	
Alpha_offset	ALPHA_OFFSET token	
Beta_offset	BETA_OFFSET token	
QP	QUANT token	
MV	MV token	
RefIdx	REF_ID token	
LFDisable	BIT token	
Output		
Name	**Token**	
BS	BS token	
DB_I	DB_SAMPLE token	
MB_OUT	MB token	
Parameter		
Name	**Description**	**Range**
LUMA_CHROM	Luma or chroma	[0,1]
QUANT_SZ	Size in bits of the QP and Scalinglist ports	[0…32]

7.3.2 Algo_DBF_AdaptiveFilter_AVC FU

FU Name	Algo_DBF_AdaptiveFilter_AVC	
Description	This FU implements deblocking filter without supporting MBAFF	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
DB_I	DB_SAMPLE token	
BS	BS token	
Output		
Name	**Token**	
DB_O	DB_SAMPLE token	
Parameter		
Name	**Description**	**Range**
LUMA_CHROME	Luma or chroma	[0,1]

7.3.3 Algo_Interp_EighthPelBilinear FU

FU Name	Algo_Interp_EighthPelBilinear	
Description	This FU performs fractional chroma sample interpolation with bilinear filter. Interpolating a sample in a fractional position requires at most 2x2 integer samples around the interpolated location.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
RD	MEM_DATA token	
PartSZ	PART_SIZE token	
Frac	FRACTION token	
Output		
Name	**Token**	
INTERP	MB token	
Parameter		
Name	**Description**	**Range**

7.3.4 Algo_Interp_SeparableSixTapQuarterPelAVC FU

FU Name	Algo_Interp_SeparableSixTapQuarterPelAVC
Description	This FU performs fractional luma sample interpolation with separable 6-tap FIR linear phase filter. Interpolating a sample in a fractional position requires at most 6x6 integer samples around the interpolated location:
Profiles@levels supported	MPEG-4 AVC Constrained BP
Input	
Name	**Token**
RD	MEM_DATA token
PartSZ	PART_SIZE token
Frac	FRACTION token

Output		
Name	**Token**	
INTERP	MB token	
Parameter		
Name	**Description**	**Range**

7.3.5 Algo_Interp_split_MB FU

FU Name	Algo_Interp_split_MB	
Description	This module reconstructs a 16x16 inter prediction macroblock from the partition of inter prediction.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
INTERP	MB token	
MbPartHeigth	PART_HEIGHT token	
MbPartIdx	PART_ID token	
MbPartWidth	PART_WIDTH token	
Output		
Name	**Token**	
MBPred	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of INTERP, MBPred ports	[0..16]

7.3.6 Algo_Interp_split_MB_C FU

FU Name	Algo_Interp_split_MB_C	
Description	This module reconstructs an 8x8 inter prediction macroblock from the partition of inter prediction.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
INTERP	MB token	
MbPartHeigth	PART_HEIGHT token	
MbPartIdx	PART_ID token	
MbPartWidth	PART_WIDTH token	
Output		
Name	**Token**	
MBPred	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of INTERP, MBPred ports	[0..16]

7.3.7 Algo_MVR_MultiFrameAdaptive FU

FU Name	Algo_MVR_MultiFrameAdptive	
Description	This FU finds a motion vector predictor by using reference index and MV of neighboring blocks (left, top, top right). This process is adaptively switched according to the partition size of the current block and the availability of the neighbouring blocks.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
mbPartIdx	MB_ID token	
PartS	COORDINATE token	
RefIdxLX	REF_ID token	
RefIdxLXN	REF_ID token	
mvLX	MV token	
Output		
Name	**Token**	
mvpLX	MV token	
Parameter		
Name	**Description**	**Range**

7.3.8 Mgnt_DPB_without_adaptiveFilter FU

FU Name	Mgnt_DPB_without_adaptiveFilter	
Description	This module stores decoded pictures input from WD port according to the macroblock address (CurrMbAddr), and sends parts of a selected stored picture output through RD port for inter prediction, according to the prediction type (Mb_Type), current position of the macroblock (CurrMbAddr), a motion vector (MV) and an ID of frame (RefIdx). This module also reorganizes and erases the picture stored into its internal memory according to RefList input.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
PicSizeInMb	SIZE token	
PicWidthInMb	WIDTH token	
RefIdx	REF_ID token	
RefList	REF_ORDER token	
CurrMbAddr	MB_ID token	
WD	MB token	
MV	MV token	
Output		
Name	**Token**	
RD	MEM_DATA token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of INTERP, MBPred ports	[0..16]
NB_PIC	Maximum number of frame stored in memory	[0..16]
MB_WIDTH	Width and Height in pixel of a macroblock	[8;16]

7.3.9 Mgnt_Buffer_Neighbor_FullMb FU

FU Name	Mgnt_Buffer_intra	
Description	This module stores bottom and right edge from macroblock receive in input MB_IN and send to EDGE according to CurrMbAddr and Mb_Type when full macroblock intra-prediction is needed.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CurrMbAddr	MB_ID token	
MB_IN	MB token	
Mb_Type	MB_TYPE token	
PicWidthInMb	WIDTH token	
Output		
Name	**Token**	
EDGE	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of data in MB_IN and EDGE ports	[0..16]
MB_WIDTH	Size in pixel of macroblocks	[8;16]

7.3.10 Mgnt_Buffer_Neighbor_4x4 FU

FU Name	Mgnt_Buffer_intra	
Description	This module stores bottom and right edge from macroblock receive in input MB_IN and send to EDGE according to CurrMbAddr and Mb_Type when 4x4 macroblock intra-prediction is needed.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CurrMbAddr	MB_ID token	
MB_IN	MB token	
Mb_Type	MB_TYPE token	
PicWidthInMb	WIDTH token	
Output		
Name	**Token**	
EDGE	MB token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of data in MB_IN and EDGE ports	[0..16]
MB_WIDTH	Size in pixel of macroblocks	[8;16]

7.3.11 Algo_MMCO

FU Name	Algo_MMCO	
Description	This module marks the index of frames to store as long term reference, short term reference, to delete or reorganize in memory. RefReordering indicates whether or not a reordering of the frame stored in memory is necessary.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
MMCO	MMCO token	
RefReordering	BIT token	
Output		
Name	**Token**	
RefList	REF_ORDER token	
Parameter		
Name	**Description**	**Range**
NB_PIC	Maximum number of frames that can be stored in memory	[0..16]

7.3.12 Mgnt_FBAddr_Chroma_MxN FU

FU Name	Mgnt_FBAddr_Chroma_MxN	
Description	This FU activate (according to value in input MB_TYPE) inter prediction issues a reference frame ID to the chroma frame buffer to fetch necessary data for the chroma fractional sample interpolation. It also notifies the chroma fractional sample interpolation FU, which mode is selected via context information Frac. It also provides the error of the current prediction.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
MV	MV token	
RefIdx	REF_ID token	
Location	COORDINATE token	
PartSZ	PART_SIZE token	
PicWidthInMb	WIDTH token	
PicSizeInMb	SIZE token	
Coef_ACR	MB token	
MB_TYPE	MB_TYPE token	
Output		
Name	**Token**	
Frac	FRACTION token	
Refbuf	REF_ID token	
Coef_ACR_P	MB token	
Parameter		
Name	**Description**	**Range**
MB_WIDTH	Width and Height in pixel of a macroblock	[8;16]
SAMPLE_SZ	Size in bits of INTERP, MBPred ports	[0..16]

7.3.13 Mgnt_Interp_FBAddr_Luma_MxN FU

FU Name	Mgnt_Interp_FBAddr_Luma_MxN	
Description	This FU issues an address and a reference frame ID to the luma frame buffer to fetch necessary data for the luma fractional sample interpolation. It also notifies the luma fractional sample interpolation FU, which mode is selected via Frac.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
MV	MV token	
RefIdx	REF_ID token	
Location	COORDINATE token	
PartSZ	PART_SIZE token	
PicWidthInMb	WIDTH token	
PicSizeInMb	SIZE token	
Coef_ACR	MB token	
MB_TYPE	MB_TYPE token	
Output		
Name	**Token**	
Frac	FRACTION token	
Refbuf	REF_ID token	
Coef_ACR_P	MB token	
Parameter		
Name	**Description**	**Range**
MB_WIDTH	Width and Height in pixel of a macroblock	[8;16]
SAMPLE_SZ	Size in bits of INTERP, MBPred ports	[0..16]

7.3.14 Mgnt_POC FU

FU Name	Mgnt_POC	
Description	This module selects the order of the frame to display according to POC input.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CurrMbAddr	MB_ID token	
POC	POC token	
MB_IN	MB token	
Output		
Name	**Token**	
Display	MB token	
Parameter		
Name	**Description**	**Range**
NB_PIC	Number of frame to store buffer display frames	[0..16]
MB_WIDTH	Width and Height in pixel of a macroblock	[8;16]
SAMPLE_SZ	Size in bits of display and MB_IN ports	[0..16]

7.3.15 Mgnt_MVR FU

FU Name	Mgnt_MVR	
Description	This module sends the information of Location, Partition size (PartSZ), motion vector associated (mvLX) and the id of reference picture (refIdxLn) for each partition within an inter predicted macroblock.	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
CurrMbAddr	MB_ID token	
MV	MVD token	
MbPartHeight	PART_HEIGHT token	
MbPartIdx	PART_ID token	
MbPartWidth	PART_WIDTH token	
PicWidthInMb	WIDTH token	
RefIdx	REF_ID token	
Output		
Name	**Token**	
LOCATION	COORDINATE token	
PartSZ	PART_SIZE token	
RefIdxLN	REF_ID token	
Parameter		
Name	**Description**	**Range**

7.3.16 Algo_Add FU

FU Name	Algo_Add	
Description	This module adds 2 Motion vectors tokens	
Profiles@levels supported	MPEG-4 AVC Constrained BP	
Input		
Name	**Token**	
X	MV token	
Output		
Name	**Token**	
Y	MV token	
Parameter		
Name	**Description**	**Range**
SAMPLE_SZ	Size in bits of the VALUE and Block ports	[0..16]

Annex A
(normative)

Naming convention of FU

A.1 Naming convention

This Annex introduces the convention used to name FUs. Each FU has a unique name in this part of ISO/IEC 23002.

The skeleton of FU name is in the following format:

{Role}_{Name}_ [property1-property2-…-propertyN]_ [size]_{STANDARD if exists}_[Ver_ID]

{…} → compulsory

[…] → optional

- → inside field separator

_ → separator between fields

Some examples:

- MGNT_Address_mpeg4_16x16
- ALGO_Interpolation_halfpel-mpeg4
- ALGO_DCRaddressing_mpeg4_8x8

A.2 Description of the fields

{Role}

This part is compulsory. It specifies if this Functional Unit implements a specific coding algorithm or not. If the FU is a video coding tool, the "ALGO" tag must be used (stands for "algorithmic content"). If not, the "MGNT" tag must be used (stands for "data management").

{Name}

1. This part is compulsory. This field corresponds to the name of the Functional Unit. It must describe as much as possible the action performed by the Functional Unit. In case of an "ALGO" type it may be related to the standard sub-clause title or to the (shorter) common name to refer to such specific algorithm. For the other Functional Unit, one would try to put a name which can be understood by the entire MPEG community.

[property]

This part is optional. It provides additional information about the Functional Unit. Several properties can be mentioned in the name.

The property field can be used to specify:

- a given characteristic of the algorithm. Example: ALGO_Interpolation_halfpel, “Halfpel” being the property field.

- luminance of chrominance appliance: “LUMA” or ”CHROMA”. Example: ALGO_Name_Luma or ALGO_Name_Chroma

- an implementation: It provides any additional information about the implementation of the Functional Unit. If there are various algorithms for one Functional Unit, this field is used to distinguish them.

[size]

This part is optional. This field gives an indication concerning the amount of data the Functional Unit has been designed for. For example the MGNT_Address_16x16 Functional Unit deals with blocks of 16x16 pixels.

{STANDARD if exists}

This tag indicates which standard the Functional Unit is conformant with.

Example: algo_Interpolation_halfpel_mpeg4 means this FU is conformant with MPEG4 Part 2. Thus, the algorithm is described in the corresponding official document.

- standard appliance: “mpeg” + {2,4,avc}. Example: “mpeg24avc”, “mpeg4avc”, “mpeg4”.

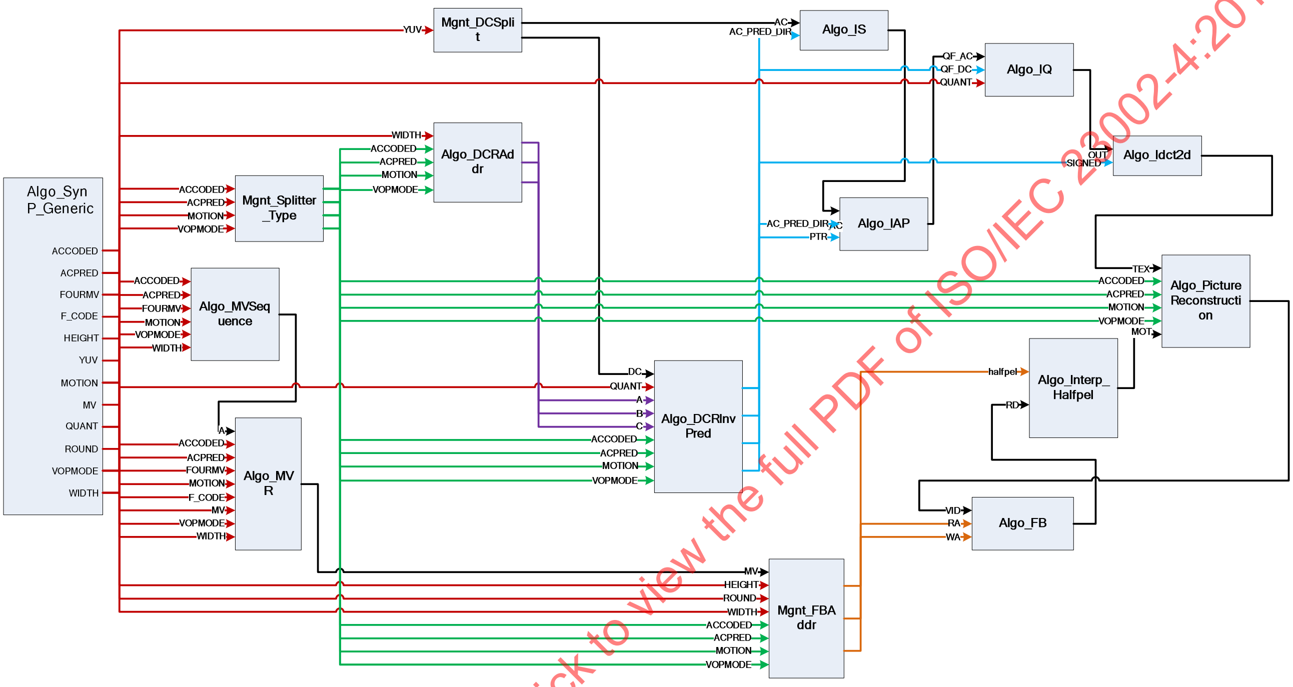
[Ver_ID]

This tag is used for revision of FU description.

Annex B
(informative)

FU Network Examples

B.1 MPEG-4 Simple Profile



A complete example is given as follows.

```
<?xml version="1.0" encoding="UTF-8"?><XDF name="Decoder">
  <!-- ************************************** -->
  <!-- Input ports of the Graph              -->
  <!-- ************************************** -->
  <!-- ************************************** -->
  <!-- Output ports of the Graph              -->
  <!-- ************************************** -->
  <!-- ************************************** -->
  <!-- Variables and Parameters of the Graph  -->
  <!-- ************************************** -->
  <Decl kind="Variable" name="ADDR_SZ">
    <Expr kind="Literal" literal-kind="Integer" value="24"/>
  </Decl>
  <Decl kind="Variable" name="FLAG_SZ">
    <Expr kind="Literal" literal-kind="Integer" value="4"/>
  </Decl>
  <Decl kind="Variable" name="MAXH_IN_MB">
    <Expr kind="Literal" literal-kind="Integer" value="69"/>
  </Decl>
  <Decl kind="Variable" name="MAXW_IN_MB">
    <Expr kind="Literal" literal-kind="Integer" value="121"/>
  </Decl>
  <Decl kind="Variable" name="MB_COORD_SZ">
    <Expr kind="Literal" literal-kind="Integer" value="8"/>
  </Decl>
  <Decl kind="Variable" name="MEM_SZ">
    <Expr kind="Literal" literal-kind="Integer" value="16"/>
  </Decl>
  <Decl kind="Variable" name="MV_SZ">
    <Expr kind="Literal" literal-kind="Integer" value="9"/>
  </Decl>
  <Decl kind="Variable" name="OUT_SZ">
    <Expr kind="Literal" literal-kind="Integer" value="10"/>
```