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)

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.

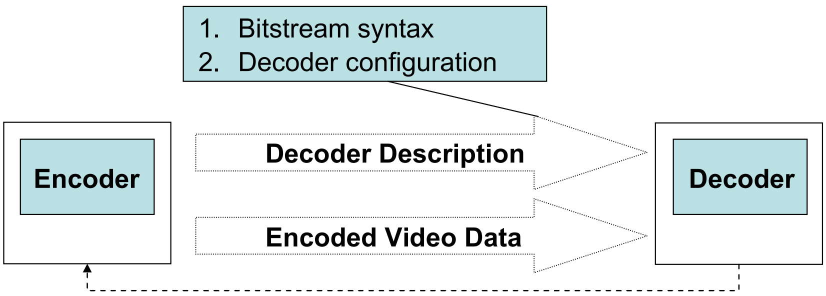

Figure 1 — Conceptual diagram of RVC

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.

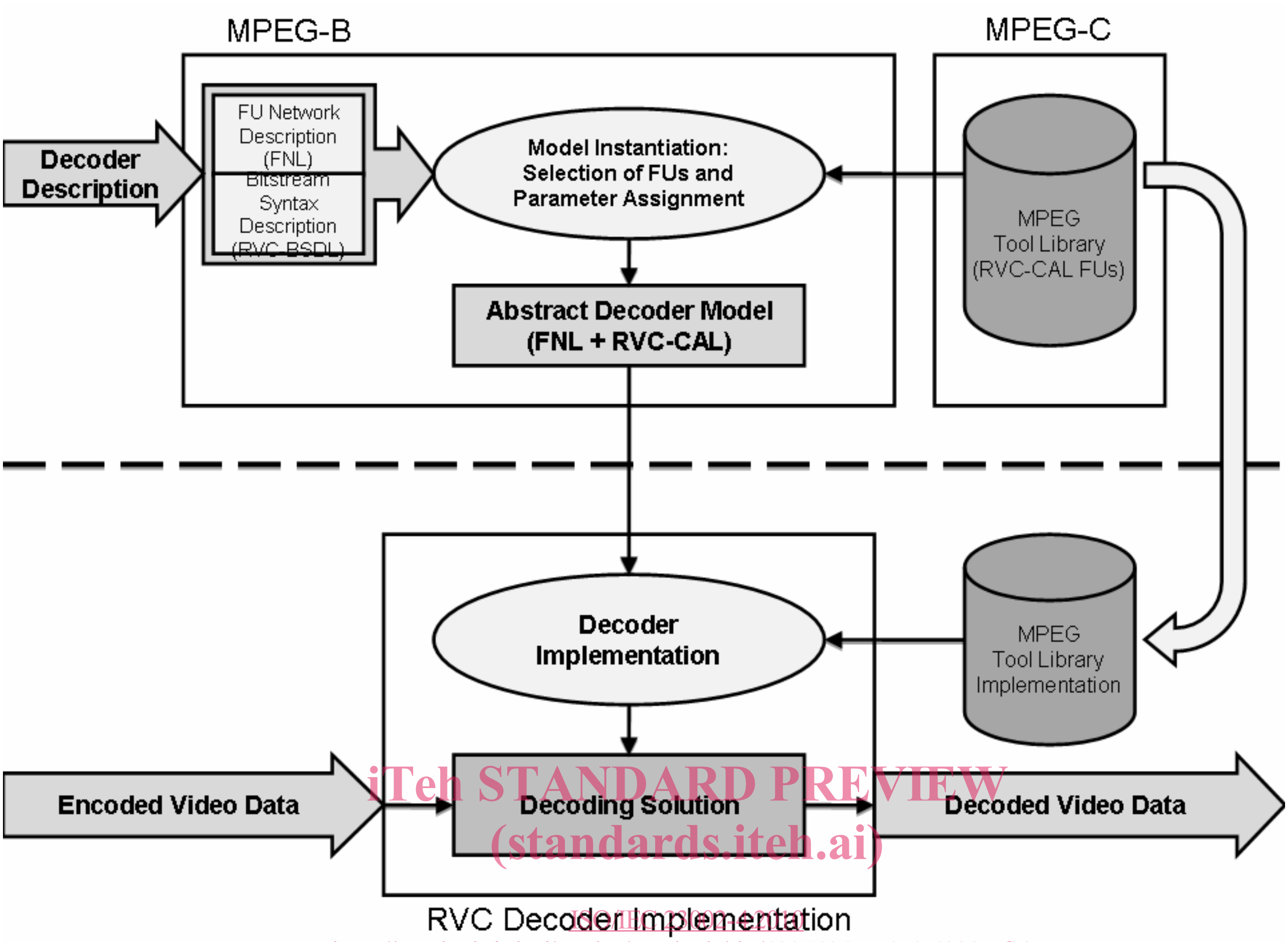

Figure 2 — Graphical representation of the process for setting up a decoding solution under the RVC framework

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	