INTERNATIONAL STANDARD

ISO/IEC 23002-4

Second edition
2014-04-15

AMENDMENT 1
2014-11-15

Information technology — MPEG video technologies —

Part 4:
Video tool library

AMENDMENT 1: Graphics tool library (GTL) for the reconfigurable multimedia coding (RMC) framework

Technologies de l'information — Technologies vidéo MPEG — Partie 4: Bibliothèque d'outils vidéo

AMENDEMENT 1: Bibliothèque d'outils graphiques (GTL) pour le cadre de codage multimédia reconfigurable (RMC)

Reference number
ISO/IEC 23002-4:2014/Amd.1:2014(E)

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

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Information technology — MPEG video technologies —

Part 4:
Video Tool Library

AMENDMENT 1: Graphics tool library (GTL) for the reconfigurable multimedia coding (RMC) framework

In 4.1 "FU Interfaces", before Table 1, add the following text:

- In several FU diagrams, the ports are named with a trailing "_i" for the input port type and with a trailing "_o" for the output port type.
- Some FU diagrams contains as well the Finite State Machine diagram. The following conventions apply: INPUT - the action of reading a token or a set of tokens from the input port, OUTPUT - the action of writing the token or a set of tokens to an output port.
- "Parameter" is set at network configuration stage (cannot be changed during the process) and it is characteristic for each FU
- Token RANGE: describes the mathematical interval for the token value

Examples:
Token RANGE: { 0, 1} – binary value
Token RANGE: [0 .. N], $value \in [0, N\ ]$ real values, closed interval

- All the FUs require the data to be in little-endian format.

In 4.2 "FU IDs", complete Table 2 with the following lines:

Note: update the FU table..

ID	FU Name
107	Algo_Parser_SC3DMC
108	Algo_InverseQuantization1D
109	Algo_InverseQuantizationND
110	Algo_InversePrediction1D
111	Algo_InversePredictionND
112	Algo_ED_AD_StaticBit
113	Algo_ED_AD_AdaptiveBit
114	Algo_ED_VLD
115	Algo_ED_BitPrecision
116	Algo_ED_AD
117	Algo_ED_AD_EG
118	Algo_ContextModeling

119	Algo_ContextModeling_SVA_nType
120	Algo_ContextModeling_SVA_Indexes
121	Algo_ContextModeling_SVA_Vertex_Attribute
122	Algo_ED_4bitsD
123	Algo_ED_ FixedLength
124	Algo_LookUpTable1D
125	Algo_DecodeConnectivity_SVA
126	Algo_DecodeConnectivity_TFAN
127	Algo_ExtractMask_SC3DMC
128	Algo_ExtractFaceDirection_SVA
129	Algo_simpleMath_2op
130	Algo_Connectivity_InversePrediction_SVA
131	Mgnt_Replicate_1_2
132	Mgnt_Replicate_1_4
133	Mgnt_Replicate_1_8
134	Mgnt_MUX_2_1
135	Mgnt_MUX_4_1
136	Mgnt_MUX_8_1
137	Mgnt_DEMUX_1_2
138	Mgnt_DEMUX_1_4
139	Mgnt_DEMUX_1_8
140	Mgnt_ExtractSegment
141	Mgnt_ProviderValue
142	Mgnt_RepeatSegment
143	Mgnt_ExtractBytes
144	Mgnt_ExtractBits
145	Mgnt_Provider1D
146	Mgnt_Provider2D

4.3 "Token Pool":

Add the following rows at the end of the table.

ID & Name	Description
55 BOOLEAN	Token which value is 0 or 1.
56 SIGN	Token which value is 0 or 1.
57 FLAG	Token which value is 0 or 1.
58 UINT_2	Unsigned integer on 2 bits.
59 UINT_4	Unsigned integer on 4 bits.
60 UINT_8	Unsigned integer on 8 bits.
61 UINT_16	Unsigned integer on 16 bits.
62 UINT_32	Unsigned integer on 32 bits.

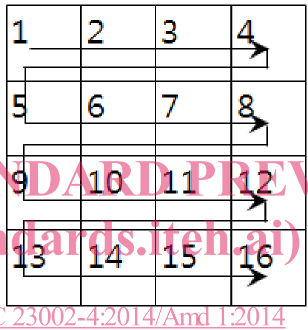
63 UINT_64	Unsigned integer on 64 bits.
64 INT_8	Integer on 8 bits.
65 INT_16	Integer on 16 bits.
66 INT_32	Integer on 32 bits.
67 INT_64	Integer on 64 bits.
68 FLOAT_32	Float on 32 bits.

Add 4.4 "Array data order":

4.4 Array data order

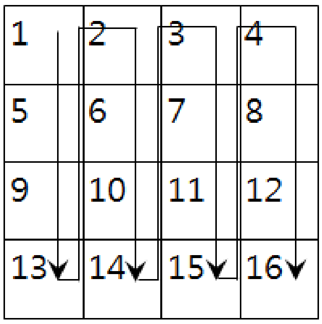
- Row based:
  - The data is processed or sent in a sequential order, row by row

    Example:



- Column based
  - The data is processed or sent in a sequential order, column by column

    Example:



Add 4.5 "Input ports":

4.5 Input ports (reset_i, init_i, start_i)

An FU does not have an outside synchronization signal or synchronization mechanism. These ports are used for the purpose of changing the values of the local variables to default or initialization values.

Add 4.6 "FU block diagram notations":

4.6 FU block diagram notations

The notation [EMBED] defines a part of the main FSM schematic that is described as a separate schematic (for complexity reasons). The [EMBED] schematic is an integrated part of the main FSM schematic

The notation [MODULE] defines a part of the main FSM schematic that is defined as a separate FU. The module schematic is integrated in the main schematic with the entire FU logic, except the “START” FSM state. The INPUT/OUTPUT states do not read or write values from the ports, they refer to local variables relative to the FU that embeds the other schematic.

Add 4.7 "Conventions":

4.7 Conventions

The significance of the “sign” port values is:

Value	Significance
0	negative
1	positive

The significance of the “flag” values is:

Value	Significance
0	false
1	true

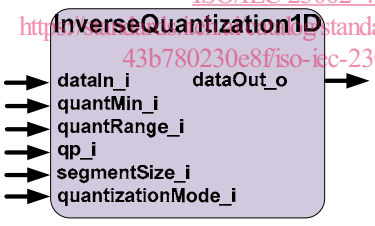
Add 5.2 "General Processing FUs":

5.2 General Processing FUs

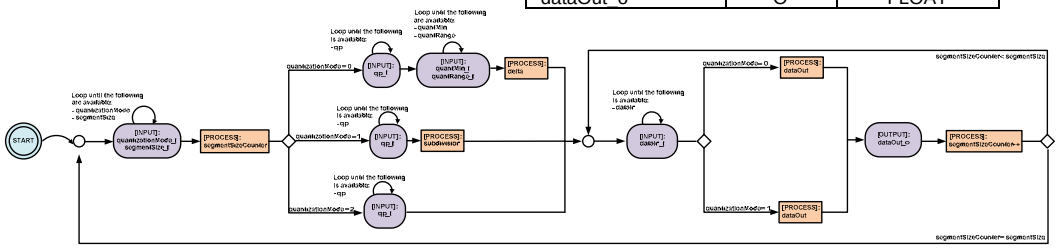
5.2.1 Algo_InverseQuantization1D

FU Name	Algo_InverseQuantization1D

Description



Port Name	Direction (I/O)	Token TYPE
dataIn_i	I	INT8, INT16, INT32, INT64
qp_i	I	UINT_32
quantMin_i	I	FLOAT
quantRange_i	I	FLOAT
segmentSize_i	I	UINT8, UINT16, UINT32, UINT64
quantizationMode_i	I	UINT_2
dataOut_o	O	FLOAT

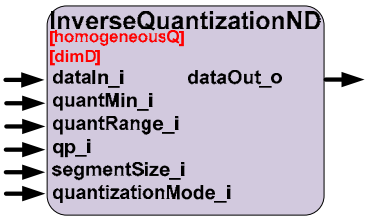
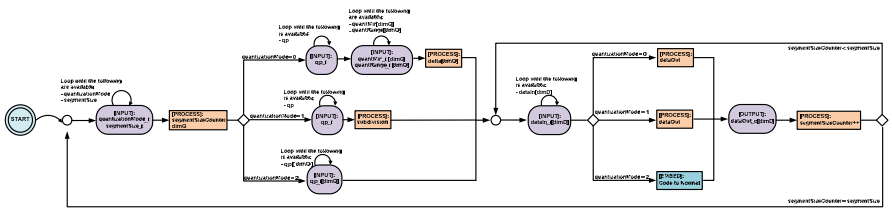


Inverse Quantization Process:

```
START:
INPUT_SEGMENT_PARAM:
INPUT:
        quantizationMode
        segmentSize
SegmentSizeCounter = 0
IF quantizatioMode = 0
    INPUT:
          qp
          quantMin
          quantRange
    IF ( quantRange > 0.0 )
          delta = ( ( 1 << qp ) - 1 ) / quantRange
```

	ELSE delta = 1.0 IF quantizationMode = 1 INPUT: qp INPUT_DATA_IN: INPUT: dataIn IF quantizatioMode = 0 PROCESS: dataOut = quantMin + (dataIn / delta) IF quantizatioMode = 1 PROCESS: dataOut = dataIn / qp OUTPUT dataOut SegmentSizeCounter ++ IF SegmentSizeCounter < segmentSize GOTO INPUT_DATA_IN ELSE GOTO INPUT_SEGMENT_PARAM The following table contains the quantization types index used in the Inverse Quantization 1D FU: **Name of quantization mode** / **Value** Uniform Quantization / **0** Uniform Texture Quantization / **1** The "Inverse Quantization" is an algorithm (step by step procedures) that allows a set of data to be represented with a limited set of values that are associated with its nearest representative. For a number of "segmentSize" of input data (dataIn), it uses the same set of quantMin, quantRange and quantValue to produce a set of output data (dataOut) of size "segmentSize"
ISO Standards using the FU	ISO/IEC 14496-16:2011
Profiles@levels supported	

5.2.2 Algo_InverseQuantizationND

FU Name	Algo_InverseQuantizationND
Description	 Port Name / Direction (I/O) / Token TYPE dataIn_i / I / INT8, INT16, INT32, INT64 qp_i / I / UINT_32 quantMin_i / I / FLOAT quantRange_i / I / FLOAT segmentSize_i / I / UINT8, UINT16, UINT32, UINT64 quantizationMode_i / I / UINT_2 dataOut_o / O / FLOAT 

Inverse Quantization Process:

$$dimQ = \begin{cases} dimD, \ if \ homogeneousQ = 0 \\ 1, \ if \ homogeneousQ = 1 \end{cases}$$

```
START:
INPUT_SEGMENT_PARAM
INPUT:
        quantizationMode
        segmentSize
SegmentSizeCounter = 0
IF quantizatioMode = 0
    INPUT:
        qp
        quantMin [ dimQ ]
        quantRange [ dimQ ]
    WHILE dimQ_counter < dimQ
        IF ( quantRange [ dimQ_counter ] > 0.0 )
            delta [ dimQ_counter ] = ( ( 1 << qp ) - 1 ) / quantRange [ dimQ_counter ]
        ELSE
            delta [ dimQ_counter ] = 1.0;
        dimQ_counter++
IF quantizationMode = 1
    INPUT:
        Qp [ dimQ ]
IF quantizationMode = 2
    INPUT:
        qp
    PROCESS:
        Subdivision = (qp - 3) / 2

INPUT_DATA_IN:
INPUT:
    dataIn [ dimQ ]
IF quantizatioMode = 0
    PROCESS:
        dataOut = quantMin [ segmentSizeCounter%dimD ] + (dataIn / delta [segmentSizeCounter%dimD])
IF quantizatioMode = 1
    PROCESS: EMBED Code 2 Normal
IF quantizatioMode = 2
    PROCESS:
        dataOut =  dataIn / qp [ segmentSizeCounter % dimD ]
OUTPUT
    dataOut
SegmentSizeCounter ++
IF SegmentSizeCounter < segmentSize
    GOTO INPUT_DATA_IN
ELSE
    GOTO INPUT_SEGMENT_PARAM


EMBED: Code 2 Normal

    Mask = ( 1 << ( 2 * subdivision) ) - 1
    tricode = data & mask;

    // Find y coordinate by solving 2nd degree equation
    factor = 1 << subdivision
    y = factor - sqrt ( (factor^2) - tricode )
    tricode = tricode + ( y * ( y - (2 * factor) ) )
    x = tricode / 2
    upsideDown = tricode % 2

    // Calculate coordinates for all vertices in triangle
    v1x = x + upsideDown
    v1y = y + upsideDown
    v2x = x + 1
    v2y = y
    v3x = x
    v3y = y + 1

    // Calculate coordinates of barycenter
```

```
invMaxCoord = 1 / factor
normal [ 0 ] = (v1x + v2x + v3x) * invMaxCoord
normal [ 1 ] = (v1y + v2y + v3y) * invMaxCoord
normal [ 2 ] = 3 - normal [ 0 ] - normal [ 1 ]

// Flip component signs if necessary
octantCode = ( data >> 2 * subdivision ) & 0x7
if  (octantCode & 0x4)
      normal [ 0 ] = (-1) * normal [ 0 ]
if (octantCode & 0x2)
      normal [ 1 ] = (-1) * normal [ 1 ]
if (octantCode & 0x1)
      normal [ 2 ] = (-1) * normal [ 2 ]

invNorm:= 1 / sqrt ( (normal[ 0 ])^2 + (normal [ 1 ])^2 + (normal [ 2 ])^2 );

//Write the 3 output values
normal [ 0 ] = normal [ 0 ] * invNorm
normal [ 1 ] = normal [ 1 ] * invNorm
normal [ 2 ] = normal [ 2 ] * invNorm

dataOut = normal
```

The following table contains the quantization types index used in the Inverse Quantization ND FU:

Name of quantization mode	Value
Uniform Quantization	**0**
Normal Quantization	**1**
Uniform Texture Quantization	**2**

The "Inverse Quantization" is an algorithm (step by step procedures) that allows a set of data to be represented with a limited set of values that are associated with its nearest representative.

For a number of “segmentSize” x “dimD” of input data (dataIn), it uses the same set of quantMin, quantRange and quantValue of size “dimD” to produce a set of output data (dataOut) of size segmentSize” x “dimD”.

For each set of size “dimD” of input data (dataIn) it uses the corresponding value of quantMin, quantRange and quantValue.

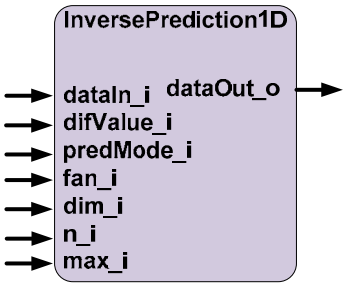
ISO Standards using the FU	ISO/IEC 14496-16:2011
Profiles@levels supported	

Parameter

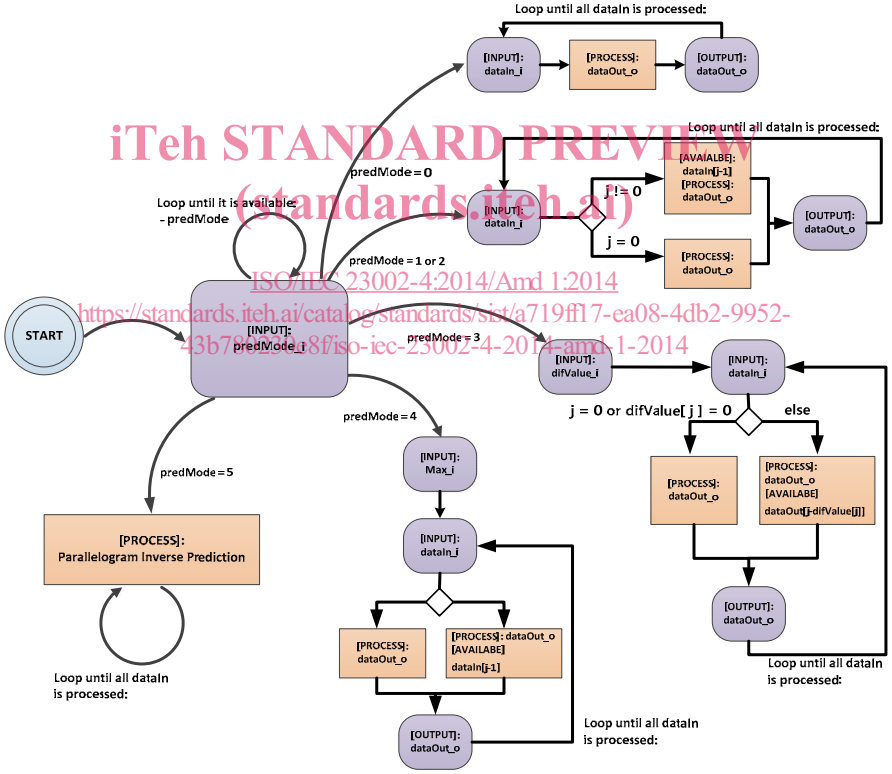
Name	Description	Type / Range
dimD	Describes the number of tokens of type dataIn_i that are consumed at each firing. This parameter is set at the network configuration level.	Type: Integer Range: [1 .. 2^5]
homogeneousQ	Describes the number of tokens of type quantRange_i, quantMin_i and quantValue_i that are necessary for the inverse quantization process. This parameter is set at the network configuration level. The number of tokens is equal to dimD if this parameter is 0 and the number of tokens is equal to 1 if this parameter is 1	Type: Boolean Range: {0,1}

5.2.3 Algo_InversePrediction1D

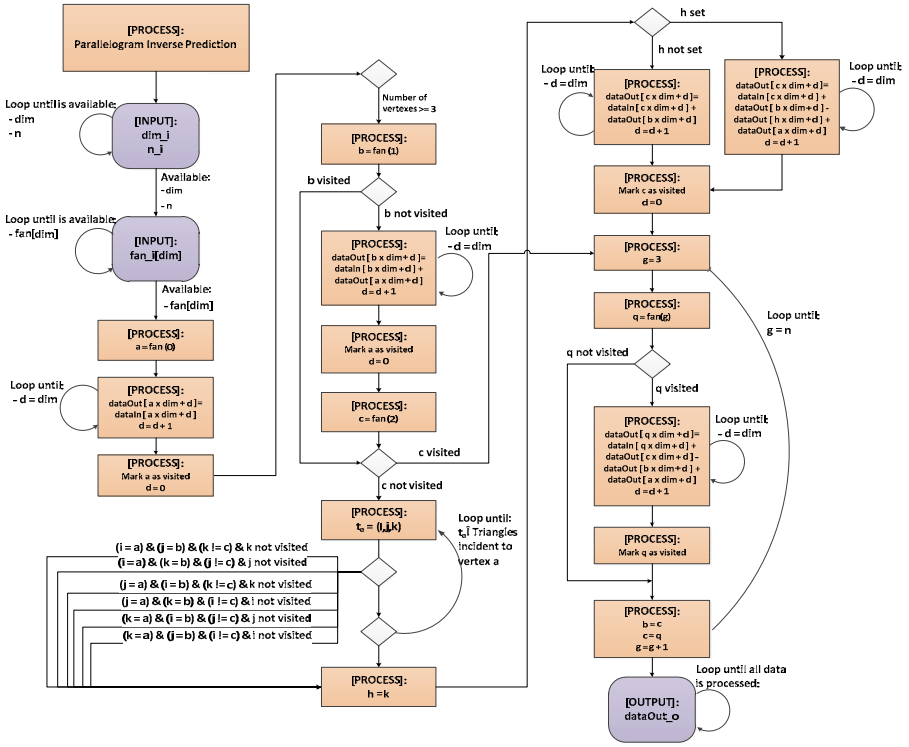
FU Name	Algo_InversePrediction1D
Description	This FU allows five modes of inverse prediction. Not all the ports are used for each prediction mode. This FU can handle one input token at a time.



Port Name	Direction (I/O)	Token TYPE
dataIn_i	I	INT8, INT16, INT32, INT64
difValue_i	I	INT8, INT16, INT32, INT64
predMode_i	I	UINT_4
fan_i	I	INT8, INT16, INT32, INT64
dim_i	I	UINT8, UINT16, UINT32, UINT64
n_i	I	INT8, INT16, INT32, INT64
max_i	I	UINT8, UINT16, UINT32, UINT64
dataOut_o	O	INT8, INT16, INT32, INT64



Process Parallelogram Inverse Prediction Schematic (FSM):



Inverse Prediction Process:

Switch (predMode)
{

Case 0: NP – No Prediction

$$dataOut[\,j\,] = dataIn\,[\,j\,],\ \forall\, j \in \{0\,..N-1\}.$$

Case 1: Diff – Differential Prediction

$$dataOut[\,j\,] = \begin{cases} DataIn\big[\,j\,\big],\ \forall\, j = 0 \\ DataIn[\,j\,] + DataOut[\,j-1\,],\ \ \forall\, j \in \{1\,..N-1\} \end{cases}$$

Case 2: XOR – based prediction

$$dataOut[\,j\,] = \begin{cases} dataIn[\,j\,],\ \forall\, j = 0 \\ dataIn\big[\,j\,\big] \otimes dataOut[\,j-1\,],\ \forall\, j \in \{1\,..N-1\} \end{cases}$$

Case 3: Adaptive Prediction

$$dataOut[\,j\,] = \begin{cases} dataIn\big[\,j\,\big], if\ difValue\big[\,j\,\big] = 0\, or\ j = 0 \\ dataIn\big[\,j\,\big] + dataOut[\,j - difValue[\,j\,]\,], otherwise \end{cases}$$

Case 4: Circular Differential Prediction

$$dataOut[\,j\,] = \begin{cases} dataIn\big[\,j\,\big],\ \forall\, j = 0 \\ d\ , if\ dataIn[\,j\,]\ < dataIn[\,j-1\,]\ \ , where \\ -d, otherwise \end{cases}$$

$$d = \begin{cases} dataIn[\,j-1\,] + M_d - dataIn[\,j\,], if\ dataIn\big[\,j\,\big] > dataIn[\,j-1\,] \\ dataIn\big[\,j\,\big] + M_d - dataIn[\,j-1\,], otherwise \end{cases}$$

Case 5: Parallelogram Inverse Prediction

```
a = fan ( 0 )
if a not visited
    d = 0
    WHILE d < dim
        dataOut [ a x dim + d ] = dataIn [ a x dim + d ]
        d = d + 1
    Mark a as visited
If number of vertexes > 3
    b = fan ( 1 )
    if b not visited
        d = 0
        WHILE d < dim
            dataOut [ b x dim + d ] = dataIn [ b x dim + d ] + dataOut  [ a x dim + d ]
            d = d + 1
        Mark b as visited
    c = fan ( 2 )
    if c not visited
        init h, t_a
        WHILE t_a = (I,j,k) ∈Triangles incident to vertex a
            If ( i=a & j=b & k≠c and k not visited ) h = k, break
            If ( i=a & k=b & j≠c and j not visited ) h = k, break
            If ( j=a & i=b & k≠c and k not visited ) h = k, break
            If ( j=a & k=b & i≠c and i not visited ) h = k, break
            If ( k=a & i=b & j≠c and j not visited ) h = k, break
            If ( k=a & b=b & i≠c and i not visited ) h = k, break
        If h not set
            d = 0
            WHILE d < dim
                dataOut [ c x dim + d ]=dataIn [ c x dim + d ]+ dataOut [ b x dim + d ]
                d = d + 1
        else
            d = 0
            WHILE d < dim
            dataOut [ c x dim + d ] =
                                dataIn [ c x dim + d ] + dataOut [ b x dim + d ]
                                – dataOut [ h x dim + d ] + dataOut [ a x dim + d ]
            d = d + 1
        Mark c as visited
    g = 3, d = 0
    WHILE g < fanSize
        q = fan ( g )
        if q not visited
            WHILE d < dim
                dataOut [ q x dim + d ] =
                            dataIn [ q x dim + d ] + dataOut [ c x dim + d ]
                            - dataOut [ b x dim + d ] + dataOut [ a x dim + d ]
                d = d + 1
            b = c
            c = q
```

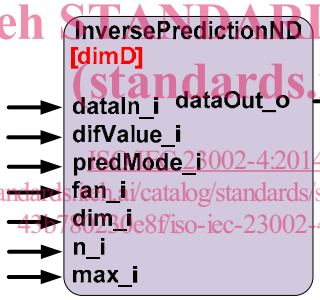

The following table contains the prediction types index used in the Inverse Prediction 1D FU:

Name of prediction mode	Value
No Prediction	**0**
Differential Prediction	**1**
XOR based prediction	**2**

	Adaptive Differential Prediction **3** Circular Differential Prediction **4** Parallelogram Inverse Prediction **5** The detailed description of the inverse parallelogram prediction is described in Annex F.
ISO Standards using the FU	ISO/IEC 14496-16:2011
Profiles@levels supported	

Parameter		
Name	**Description**	**Range**

5.2.4 Algo_InversePredictionND

FU Name	Algo_InversePredictionND
Description	This FU allows five modes of inverse prediction. Not all the ports are used for each prediction mode. This FU can handle a number of dimD input tokens at a time.  Process Schematic (FSM):

Port Name	Direction (I/O)	Token RANGE
dataIn_i	I	INT8, INT16, INT32, INT64
difValue_i	I	INT8, INT16, INT32, INT64
predMode_i	I	UINT_4
fan_i	I	INT8, INT16, INT32, INT64
dim_i	I	UINT8, UINT16, UINT32, UINT64
n_i	I	INT8, INT16, INT32, INT64
max_i	I	UINT8, UINT16, UINT32, UINT64
dataOut_o	O	INT8, INT16, INT32, INT64