DRAFT AMENDMENT ISO/IEC 23002-4:2010/DAM 3

ISO/IEC JTC **1** Secretariat: **ANSI**

Voting begins on **2013-06-25** Voting terminates on **2013-09-25**

Information technology — MPEG video technologies —

Part 4: Video tool library

AMENDMENT 3: 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 3: Bibliothèque d'outils graphiques (GTL) pour le cadre de codage multimédia reconfigurable (RMC)

ICS 35.040

To expedite distribution, this document is circulated as received from the committee secretariat. ISO Central Secretariat work of editing and text composition will be undertaken at publication stage.

Pour accélérer la distribution, le présent document est distribué tel qu'il est parvenu du secrétariat du comité. Le travail de rédaction et de composition de texte sera effectué au Secrétariat central de l'ISO au stade de publication.

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.

Amendment 3 to ISO/IEC 23002-4:2010 was prepared by Joint Technical Committee 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 3: 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..

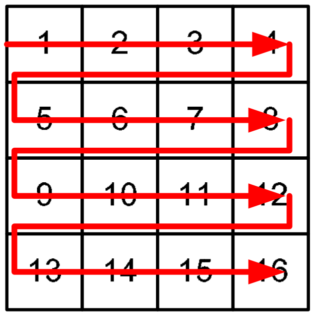
ID	FU Name
67	Algo_Parser_SC3DMC
68	Algo_InverseQuantization1D
69	Algo_InverseQuantizationND
70	Algo_InversePrediction1D
71	Algo_InversePredictionND
72	Algo_ED_AD_StaticBit
73	Algo_ED_AD_AdaptiveBit
74	Algo_ED_VLD
75	Algo_IDCT
76	Algo_ConnectivityDecoding_SVA
77	Algo_DecodeConnectivity_TFAN
78	Algo_LookUpTable
79	Algo_InverseLookUpTable
80	Algo_Matrix2Vector1D
81	Algo_Matrix2VectorND
82	Algo_Interpolate3DVector

83	Algo_Normalize4DVector
84	Algo_ConjugateQuaternion
85	Algo_ScaleQuaternion
86	Algo_ED_AD
87	Algo_ED_AD_EG
88	Algo_ED_BitPrecision
89	Algo_Decode_Symbol
90	Algo_ArithmeticDecoder_Vector
91	Algo_Decode_P_Frame
92	Algo_ED_4bitsD
93	Algo_ED_FixedLength
94	Algo_ContextModeling_nType
95	Algo_ContextModeling_SVA_INDEXES
96	Algo_ContextModeling_SVA_VERTEX_ATTRIBUTE
97	Mgnt_Replicate_1_2
98	Mgnt_Replicate_1_4
99	Mgnt_Replicate_1_8
100	Mgnt_MUX_2_1
101	Mgnt_MUX_4_1
102	Mgnt_MUX_8_1
103	Mgnt_DEMUX_1_2
104	Mgnt_DEMUX_1_4
105	Mgnt_DEMUX_1_8
106	Mgnt_ExtractSegment
107	Algo_ContextModeling_TFAN
108	Algo_ExtractMask_SC3DMC
109	Algo_ExtractFaceDirection
110	Mgnt_ProviderValue
111	Mgnt_RepeatSegment
112	Mgnt_ExtractBytes

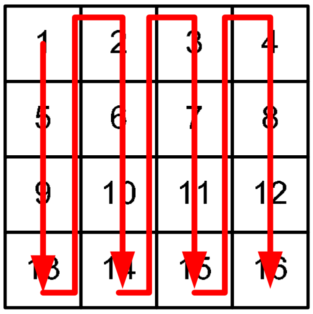
Add 4.4 "Array data order":

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 5.2 "General Processing FUs":

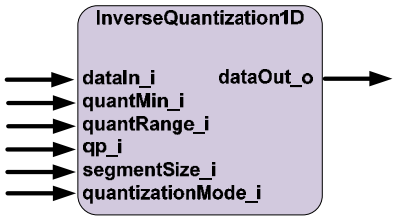
5.2 General Processing FUs

5.2.1 Algo_InverseQuantization1D

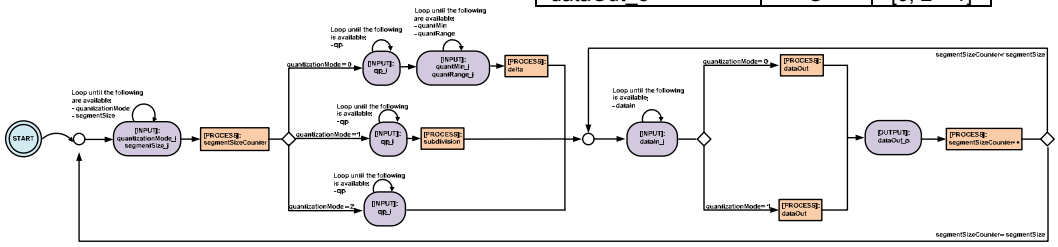
FU Name

Algo_InverseQuantization1D

Description



Port Name	Direction (I/O)	Token RANGE
dataIn_i	I	[0, 2^{32}-1]
qp_i	I	[0, 2^{32}-1]
quantMin_i	I	[0, 2^{32}-1]
quantRange_i	I	[0, 2^{32}-1]
segmentSize_i	I	[0, 2^{32}-1]
quantizationMode_i	I	[0, 2^{32}-1]
dataOut_o	O	[0, 2^{64}-1]



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 "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”

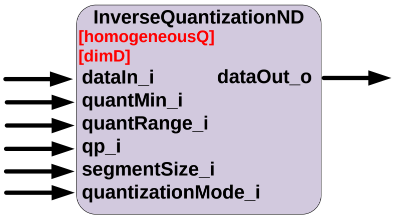
ISO Standards

ISO/IEC 14496-16:2011

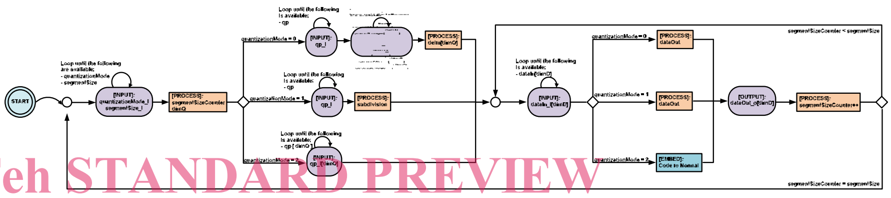
using the FU	
Profiles@levels supported	

5.2.2 Algo_InverseQuantizationND

FU Name	Algo_InverseQuantizationND
Description	(see below)



Port Name	Direction (I/O)	Token RANGE
dataIn_i	I	$[0, 2^{32}-1]$
qp_i	I	$[0, 2^{32}-1]$
quantMin_i	I	$[0, 2^{32}-1]$
quantRange_i	I	$[0, 2^{32}-1]$
segmentSize_i	I	$[0, 2^{32}-1]$
quantizationMode_i	I	$[0, 2^{8}-1]$
dataOut_o	O	$[0, 2^{64}-1]$



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 "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.

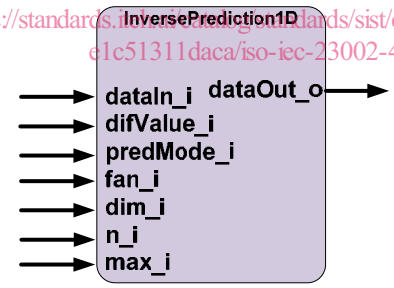
For a number of “segmentSize” x “dimD” of input data (dataIn), it uses the same set of quantMin, quantRange and quantValue of size “dimD” to produce a set of output data (dataOut) of size segmentSize” x “dimD”.

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

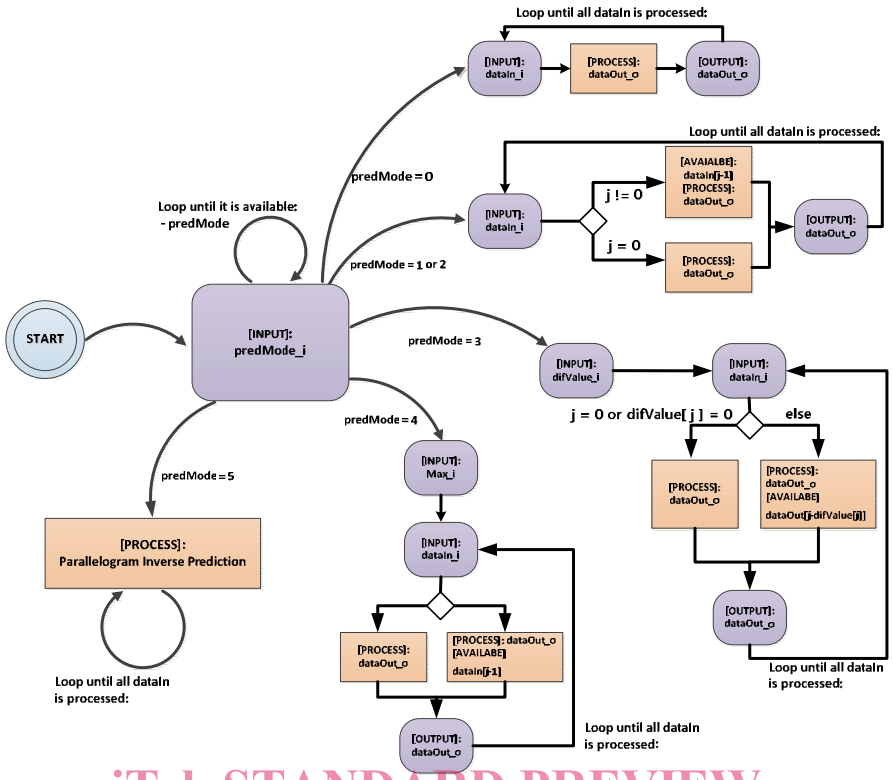
ISO Standards using the FU	ISO/IEC 14496-16:2011	
Profiles@levels supported		
Parameter		
Name	**Description**	**Type / Range**
dimD	Describes the number of tokens of type dataIn_i that are consumed at each firing. This parameter is set at the network configuration level.	Type: Integer Range: [1 .. 2^5]
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

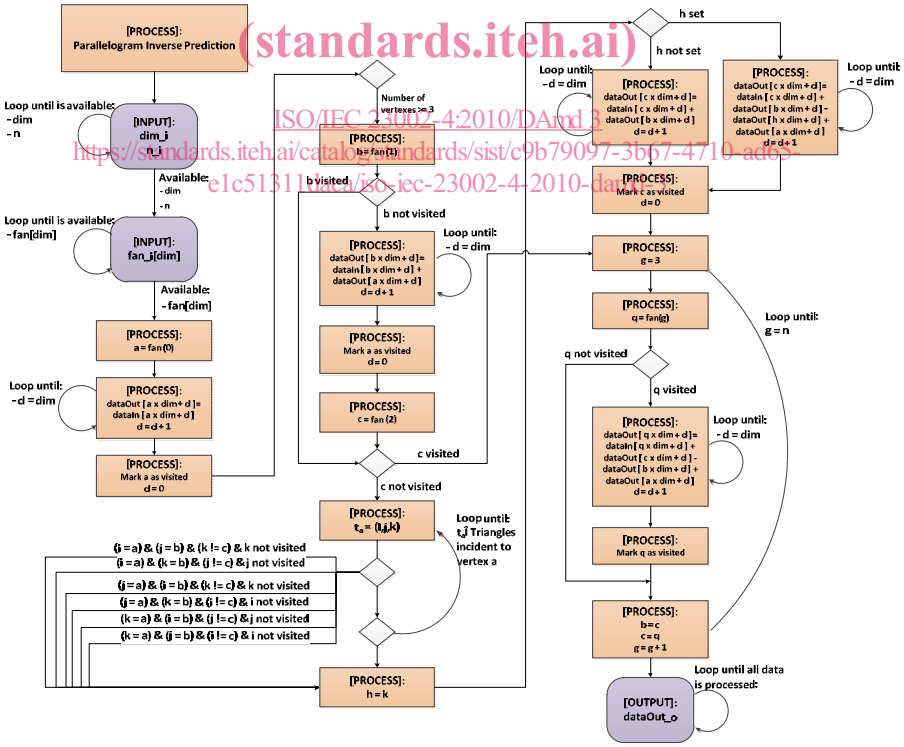
FU Name	Algo_InversePrediction1D
Description	This FU allows five modes of inverse prediction. Not all the ports are used for each prediction mode. This FU can handle one input token at a time.



Port Name	Direction (I/O)	Token RANGE
dataIn_i	I	[0, 2^{32}-1]
difValue_i	I	[0, 2^{32}-1]
predMode_i	I	[0, 2^{32}-1]
fan_i	I	[0, 2^{32}-1]
dim_i	I	[0, 2^{32}-1]
n_i	I	[0, 2^{32}-1]
max_i	I	[0, 2^{32}-1]
dataOut_o	O	[0, 2^{32}-1]



Process Parallelogram Inverse Prediction Schematic (FSM):



Inverse Prediction Process:

Switch (predMode)
{

Case 0: NP – No Prediction

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

Case 1: Diff – Differential Prediction

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

Case 2: XOR – based prediction

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

Case 3: Adaptive Prediction

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

Case 4: Circular Differential Prediction

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

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

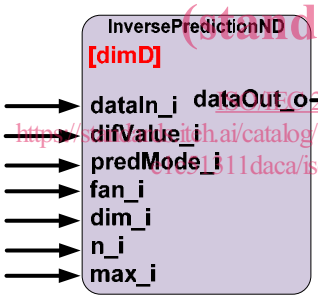
Case 5: Parallelogram Inverse Prediction

```
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, ta
        WHILE ta = (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 )
```

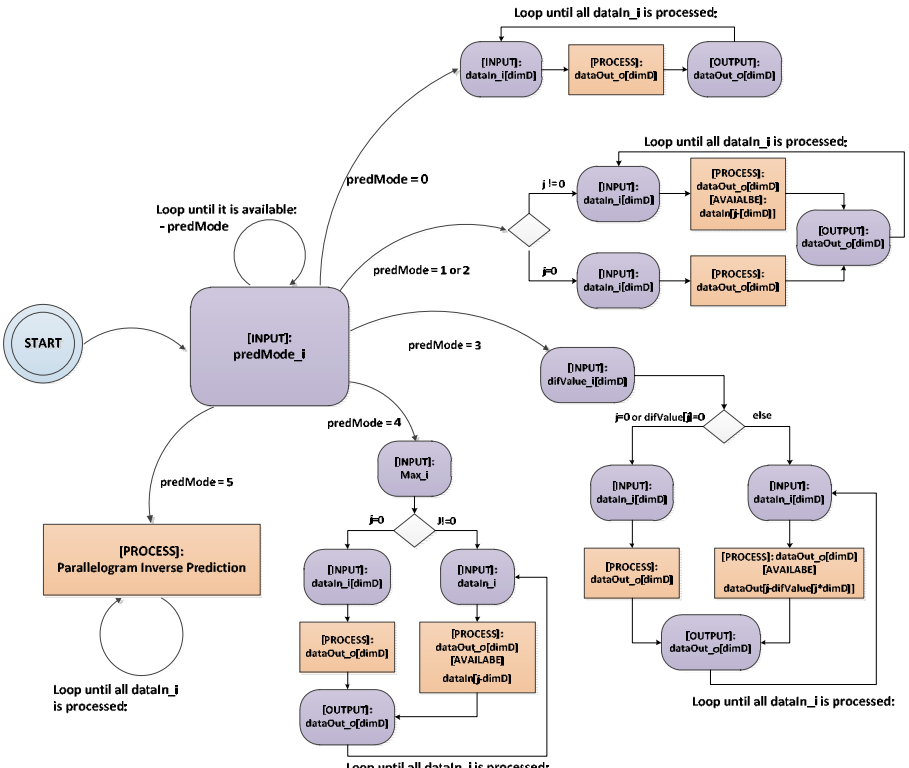
	if q not visited WHILE d < dim dataOut [q x dim + d] = dataIn [q x dim + d] + dataOut [c x dim + d] - dataOut [b x dim + d] + dataOut [a x dim + d] d = d + 1 b = c c = q The detailed description of the inverse parallelogram prediction is described in Annex E.
ISO Standards using the FU	ISO/IEC 14496-16:2011
Profiles@levels supported	

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

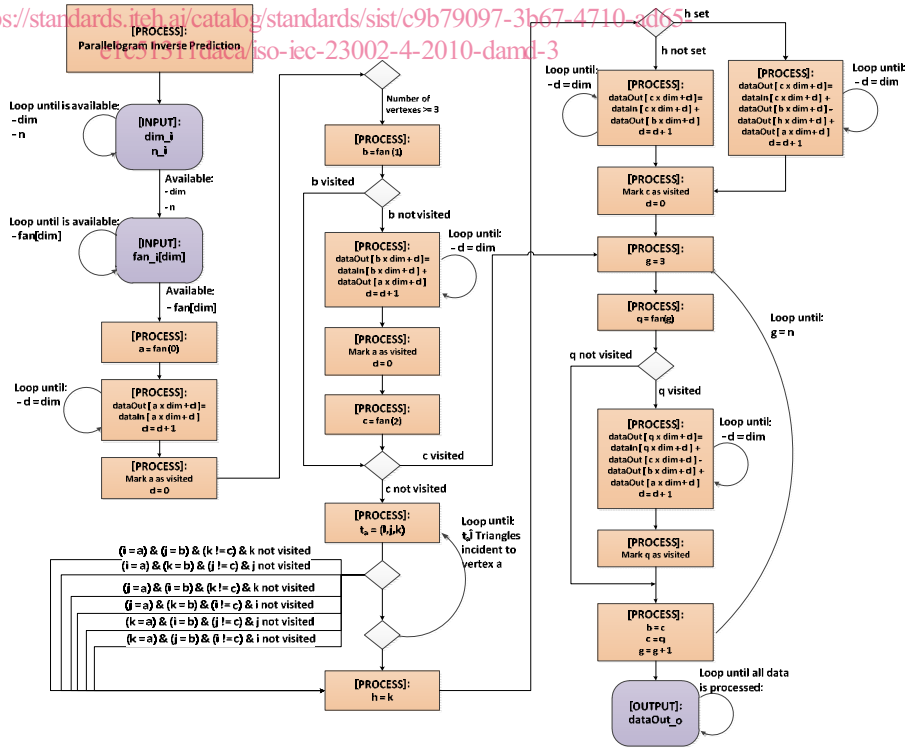
5.2.4 Algo_InversePredictionND

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	[0, 2^{32}-1]
difValue_i	I	[0, 2^{32}-1]
predMode_i	I	[0, 2^{32}-1]
fan_i	I	[0, 2^{32}-1]
dim_i	I	[0, 2^{32}-1]
n_i	I	[0, 2^{32}-1]
max_i	I	[0, 2^{32}-1]
dataOut_o	O	[0, 2^{32}-1]



Process Parallelogram Inverse Prediction Schematic (FSM):



Inverse Prediction ND Process:

Switch(predMode)