#### 7.3.8.5 Coding unit syntax

|  |  |
| --- | --- |
| coding\_unit( x0, y0, log2CbSize ) { | Descriptor |
| if( transquant\_bypass\_enabled\_flag ) |  |
| **cu\_transquant\_bypass\_flag** | ae(v) |
| if( slice\_type != I ) |  |
| **cu\_skip\_flag**[ x0 ][ y0 ] | ae(v) |
| nCbS = ( 1  <<  log2CbSize ) |  |
| if( cu\_skip\_flag[ x0 ][ y0 ] ) |  |
| prediction\_unit( x0, y0, nCbS, nCbS ) |  |
| else { |  |
| **intra\_bc\_flag**[ x0 ][ y0 ] | ae(v) |
| if( intra\_bc\_flag[ x0 ][ y0 ] ) { |  |
| **intra\_bc\_abs\_db\_greater0\_flag[ 0 ]** | ae(v) |
| **intra\_bc\_abs\_db\_greater0\_flag[ 1 ]** | ae(v) |
| if(intra\_bc\_abs\_db\_greater0\_flag[ 0 ] ) { |  |
| **intra\_bc\_abs\_db\_minus1[ 0 ]** | ae(v) |
| **intra\_bc\_db\_sign\_flag[ 0 ]** | ae(v) |
| } |  |
| if(intra\_bc\_abs\_db\_greater0\_flag[ 1 ] ) { |  |
| **intra\_bc\_abs\_db\_minus1**[ 1 ] | ae(v) |
| **intra\_bc\_db\_sign\_flag**[ 1 ] | ae(v) |
| } else { |  |
| if( slice\_type != I ) |  |
| **pred\_mode\_flag** | ae(v) |
| if( CuPredMode[ x0 ][ y0 ] != MODE\_INTRA | | log2CbSize = = MinCbLog2SizeY ) |  |
| **part\_mode** | ae(v) |
| if( CuPredMode[ x0 ][ y0 ] = = MODE\_INTRA ) { |  |
| … |  |
| } |  |
| } |  |
| if( !pcm\_flag[ x0 ][ y0 ] ) { |  |
| if( ( CuPredMode[ x0 ][ y0 ] != MODE\_INTRA &&   !( PartMode = = PART\_2Nx2N && merge\_flag[ x0 ][ y0 ] ) ) || CuPredMode[ x0 ][ y0 ] == MODE\_INTRA\_BC ) |  |
| **rqt\_root\_cbf** | ae(v) |
| … |  |
| } |  |
| } |  |
| } |  |

**7.4.9.5 Coding unit semantics**

**…**

**intra\_bc\_flag[ x0 ][ y0 ]** equal to 1 specifies that the current coding unit is coded in intra block copying mode. intra\_bc\_flag[ x0 ][ y0 ]equal to 0 specifies that the current coding unit is coded using MODE\_INTRA or MODE\_INTER

**intra\_bc\_abs\_db\_greater0\_flag**[ compIdx ] specifies whether the absolute value of a displacement vector component is greater than 0.

**intra\_bc\_abs\_db\_minus1**[ compIdx ] plus 1 specifies the absolute value of a displacement vector component.

When intra\_bc\_abs\_db\_minus1[ compIdx ] is not present, it is inferred to be equal to −1.

**intra\_bc\_db\_sign\_flag**[ compIdx ] specifies the sign of a displacement vector component as follows:

– If intra\_bc\_db\_sign\_flag[ compIdx ] is equal to 0, the corresponding displacement vector component has a positive value.

– Otherwise (intra\_bc\_db\_sign\_flag[ compIdx ] is equal to 1), the corresponding displacement vector component has a negative value.

When intra\_bc\_db\_sign\_flag[ compIdx ] is not present, it is inferred to be equal to 0.

The displacement vector difference lDb[ compIdx ] for compIdx = 0..1 is derived as follows:

lDb[ compIdx ]=intra\_bc\_abs\_db\_greater0\_flag[ compIdx ]\*( intra\_bc\_abs\_db\_minus1[ compIdx ] + 1 )\*( 1 − 2 \* intra\_bc\_db\_sign\_flag[ compIdx ] )

**9.3.2.2 Initialization process for context variables**

**Table 4‑1 – Association of ctxIdx and syntax elements for each initializationType in the initialization process**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Syntax element | ctxIdxTable | initType | | |
| 0 | 1 | 2 |
| coding\_unit() | intra\_bc\_flag | Table 9-xx | 0..2 | 3..5 | 6..8 |

**Table 9-xx. Values of variable initValue for intra\_bc\_flag ctxIdx**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Initialization variable | intra\_bc\_flag ctxIdx | | | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| initValue | 185 | 185 | 201 | 197 | 197 | 185 | 197 | 197 | 185 |

**9.3.3 Binarization process**

**9.3.3.1 General**

**Table 4‑2 – Syntax elements and associated binarizations**

| Syntax structure | Syntax element | Binarization | |
| --- | --- | --- | --- |
| Process | Input parameters |
| coding\_uint( ) | intra\_bc\_flag | FL | cMax = 1 |

**9.3.4.2 Derivatino process for ctxTable, ctxIdx and bypassFlag**

**9.3.4.2.1 General**

**Table 4‑3 – Assignment of ctxInc to syntax elements with context coded bins**

| Syntax element | binIdx | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | >= 5 |
| intra\_bc\_flag | 0,1,2 | na | na | na | na | na |

**9.3.4.2.2 Derivation process of ctxInc using left and above syntax elements**

**Table 4‑4 – Specification of ctxInc using left and above syntax elements**

|  |  |  |  |
| --- | --- | --- | --- |
| Syntax element | condL | condA | ctxInc |
| intra\_bc\_flag[ x0 ][ y0 ] | intra\_bc\_flag [ xNbL ][ yNbL ] | intra\_bc\_flag [ xNbA ][ yNbA ] | ( condL  &&  availableL ) + ( condA  &&  availableA ) |