**ISO 14496-3:2019 AMD1(X)**

ISO/IEC JTC1/SC 29/WG 6

Date: 2025-10-11

**Information technology — Coding of audio-visual objects — Part 3: Audio**

**Amendment 1: Media authenticity and immersive interchange format**

DAM stage

**Warning for WDs and CDs**

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

*A model document of an International Standard (the Model International Standard) is available at:*[*https://www.iso.org/drafting-standards.html*](https://www.iso.org/drafting-standards.html)

© ISO 20XX

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO’s member body in the country of the requester.

ISO copyright office

CP 401 • Ch. de Blandonnet 8

CH-1214 Vernier, Geneva

Phone: +41 22 749 01 11

Email: copyright@iso.org

Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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 ISO documents 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](https://www.iso.org/directives-and-policies.html)).

ISO draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO takes no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO *[had/had not]* received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at [www.iso.org/patents](http://www.iso.org/patents). ISO shall not be held responsible for identifying any or all such patent rights.

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

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see [www.iso.org/iso/foreword.html](https://www.iso.org/foreword-supplementary-information.html).

This document was prepared by Technical Committee *[or Project Committee]* ISO/TC *[or ISO/PC]* ###, *[name of committee]*, Subcommittee SC ##, *[name of subcommittee]*.

This second/third/… edition cancels and replaces the first/second/… edition (ISO #####:####), which has been technically revised.

The main changes are as follows:

— xxx xxxxxxx xxx xxxx

A list of all parts in the ISO ##### series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at [www.iso.org/members.html](https://www.iso.org/members.html).

Replace all occurrences of “ISO/IEC 14496-5” with the following:

ISO/IEC 14496-35

Add the following at the end of Section 1.2:

IETF RFC 3986, Uniform Resource Identifier (URI): Generic Syntax

IETF RFC 9562, Universally Unique IDentifiers (UUIDs)

ISO/IEC 23091-3, Information technology — Coding-independent code-points — Part 3: Audio

Replace the last 3 rows of Table 1.1 with the following

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 46 | Audio Sync |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 47-63 | Reserved for 14496-3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 64-95 | Reserved for 23003-3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Replace the last 10 rows of Table 1.17 with the following:

|  |  |  |
| --- | --- | --- |
| 0x5E | AAC Profile | L9 |
| 0x5F | ALS Simple Profile | L5 |
| 0x60 | ALS Simple Profile | L6 |
| 0x61 | ALS Simple Profile | L7 |
| 0x62 - 0x6F | reserved for ISO/IEC 14496-3 | - |
| 0x70 - 0x7F | reserved for ISO/IEC 23003-3 | - |
| 0x80 - 0xFD | user private | - |
| 0xFE | no audio profile specified | - |
| 0xFF | no audio capability required | - |
| NOTE Usage of the value 0xFE indicates that the content described by this InitialObjectDescriptor does not comply to any audio profile specified in ISO/IEC 14496-3. Usage of the value 0xFF indicates that none of the audio profile capabilities are required for this content. | | |

After “case 46” in Table 1.19 add the following:

Table 1.19 – Syntax of AudioSpecificConfig()

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| AudioSpecificConfig () |  |  |
| { |  |  |
| … |  |  |
| case 46: |  |  |
| AudioSyncFeatureSpecificConfig(): |  |  |
| break; |  |  |
| case 47: |  |  |
| iif\_specific\_config(): |  |  |
| break; |  |  |
| case 64: … case 95: |  |  |
| UsacAotSpecificConfig(audioObjectType); |  |  |
| break; |  |  |
| default: |  |  |
| /\* reserved \*/ |  |  |
| } |  |  |
| … |  |  |
| a In the Baseline USAC profile defined in ISO/IEC 23003-3, the backwards compatible signaling of SBR, PS, MPS, or SAOC at the end of the AudioSpecificConfig() (i.e., using the extensionIdentifier bitstream element) is not permitted. | | |

After 1.6.2.1.22, add the following:

1.6.2.1.23 **iif\_specific\_config**

Defined in subpart 4.

1.6.2.1.24 **UsacAotSpecificConfig**

Defined in ISO/IEC 23003-3.

Add the following at the end of Table 1.21:

|  |  |  |  |
| --- | --- | --- | --- |
| 47 | AAC IIF | ISO/IEC 14496‑3 Subpart 4 | see subclause 1.6.2.2.2.6 |

After 1.6.2.2.2.5, add the following:

**1.6.2.2.2.6 AAC IIF**

One top level payload (iif\_access\_unit()) is mapped into one access unit. A sequence of access units forms one elementary stream.

In 4.4.2.7, replace Table 4.59 with the following:

Table 4.59 – Syntax of extension\_payload()

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| extension\_payload(cnt) |  |  |
| { |  |  |
| **extension\_type**; | **4** | **uimsbf** |
| align = 4; |  |  |
| switch( extension\_type ) { |  |  |
| case EXT\_DYNAMIC\_RANGE: |  |  |
| return dynamic\_range\_info(); |  |  |
| case EXT\_UNI\_DRC: |  |  |
| return uniDrc(); |  |  |
| case EXT\_AUTH\_DATA: |  |  |
| **fill\_nibble**; /\* shall be ‘0000’ \*/ | **4** | **uimsbf** |
| return auth\_extension(); |  |  |
| case EXT\_ADD\_TYPES: |  |  |
| **extension\_type\_add**; | **4** | **uimsbf** |
| extension\_type = 0xF + extension\_type\_add; |  |  |
| switch( extension\_type ) { |  |  |
| default: |  |  |
| return 1; |  |  |
| } |  |  |
| case EXT\_SAC\_DATA: |  |  |
| return sac\_extension\_data(cnt); |  |  |
| case EXT\_SAOC\_DATA: |  |  |
| return saoc\_extension\_data(cnt); |  |  |
| case EXT\_LDSAC\_DATA: |  |  |
| return ldsac\_extension\_data(cnt); |  |  |
| case EXT\_SBR\_DATA: |  |  |
| return sbr\_extension\_data(id\_aac, 0); |  | Note 1 |
| case EXT\_SBR\_DATA\_CRC: |  |  |
| return sbr\_extension\_data(id\_aac, 1); |  | Note 1 |
| case EXT\_SAOC\_DE\_DATA: |  |  |
| return saoc\_de\_extension\_data(cnt); |  |  |
| case EXT\_DATA\_LENGTH: |  |  |
| hlp = 1; |  |  |
| **len**; | **4** | **uimsbf** |
| if (len==15) { |  |  |
| len += **add\_len;** | **8** | **uimsbf** |
| hlp += 1; |  |  |
| If (add\_len==255) { |  |  |
| len += **add\_add\_len**; | **16** | **uimsbf** |
| hlp += 2; |  |  |
| }  } |  |  |
| return hlp+extension\_payload(len); |  | **Note 2** |
| case EXT\_FILL\_DATA: |  |  |
| **fill\_nibble**; /\* shall be ‘0000’ \*/ | **4** | **uimsbf** |
| for (i=0; i<cnt-1; i++) { |  |  |
| **fill\_byte[i]**; /\* shall be ‘10100101’ \*/ | **8** | **uimsbf** |
| } |  |  |
| return cnt; |  |  |
| case EXT\_DATA\_ELEMENT: |  |  |
| **data\_element\_version**; | **4** | **uimsbf** |
| switch( data\_element\_version ) { |  |  |
| case ANC\_DATA: |  |  |
| loopCounter = 0; |  |  |
| dataElementLength = 0; |  |  |
| do { |  |  |
| **dataElementLengthPart**; | **8** | **uimsbf** |
| dataElementLength += dataElementLengthPart; |  |  |
| loopCounter++; |  |  |
| } while (dataElementLengthPart == 255); |  |  |
| for (i=0; i<dataElementLength; i++) { |  |  |
| **data\_element\_byte[i]**; | **8** | **uimsbf** |
| } |  |  |
| return (dataElementLength+loopCounter+1); |  |  |
| default: |  |  |
| align = 0; |  |  |
| } |  |  |
| case EXT\_FIL: |  |  |
| default: |  |  |
| for (i=0; i<8\*(cnt-1)+align; i++) { |  |  |
| **other\_bits[i]**; | **1** | **uimsbf** |
| } |  |  |
| return cnt; |  |  |
| } |  |  |
| } |  |  |
| Note 1: id\_aac is the id\_syn\_ele of the corresponding AAC element (ID\_SCE or ID\_CPE) or ID\_SCE in case of CCE.  Note 2: The extension\_payload() included here shall not have extension\_type == EXT\_DATA\_LENGTH. | | |

Add the following after 4.4.2.9:

* + - 1. **Payloads for the audio object type AAC IIF**

**Table 4.87 – Syntax of iif\_access\_unit()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_acess\_unit() |  |  |
| { |  |  |
| while (bits\_to\_decode() > 23) { |  |  |
| iif\_block(); |  |  |
| } |  |  |
| } |  |  |

**Table 4.88 – Syntax of iif\_block()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_block() |  |  |
| { |  |  |
| iif\_block\_header(); |  |  |
| iif\_block\_payload(); |  |  |
| iif\_block\_footer(); |  |  |
| } |  |  |

**Table 4.89 – Syntax of iif\_block\_header()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_block\_header() |  |  |
| { |  |  |
| **frame\_counter;** | **4** | **uimsbf** |
| **block\_id;** | **5** | **uimsbf** |
| **block\_priority;** | **2** | **uimsbf** |
| **block\_size;** | **11** | **uimsbf** |
| **b\_block\_protected;** | **1** | **uimsbf** |
| **reserved;** | **1** | **uimsbf** |
| block\_bytes\_left = block\_size; |  |  |
| if (block\_size == 0x7ff) { |  |  |
| ext\_block\_size = 0; |  |  |
| ext\_blk\_size\_loop\_cnt = 0; |  |  |
| b\_more\_bits = 1; |  |  |
| while (0 != b\_more\_bits) { |  |  |
| if (ext\_blk\_size\_loop\_cnt < 2) { |  |  |
| **block\_size\_extension**; | **7** | **uimbsf** |
| **b\_more\_bits**; | **1** | **uimsbf** |
| ext\_block\_size <<= 7; |  |  |
| } else { |  |  |
| **block\_size\_extension**; | **8** | **uimsbf** |
| b\_more\_bits = 0; |  |  |
| ext\_block\_size <<= 8; |  |  |
| } |  |  |
| ext\_block\_size += block\_size\_extension; |  |  |
| ext\_blk\_size\_loop\_cnt += 1; |  |  |
| } |  |  |
| block\_size += ext\_block\_size + ext\_blk\_size\_loop\_cnt; |  |  |
| block\_bytes\_left = block\_size – ext\_blk\_size\_loop\_cnt; |  |  |
| } |  |  |
| if (b\_block\_protected) { |  |  |
| block\_bytes\_left = block\_bytes\_left - 2; |  |  |
| } |  |  |
| } |  |  |

**Table 4.90 – Syntax of iif\_block\_payload()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_block\_payload() |  |  |
| { |  |  |
| samplingFrequency = block\_sampling\_rate[block\_id]; |  |  |
|  |  |  |
| /\* Config Update \*/ |  |  |
| **b\_config\_update\_inband;** | **1** | **uimsbf** |
| if (b\_config\_update\_inband) { |  |  |
| iif\_config\_update(0); |  |  |
| } |  |  |
| else { |  |  |
| **dynamic\_config\_upd\_idx;** | **3** | **uimsbf** |
| } |  |  |
|  |  |  |
| iif\_metadata\_payload(); |  |  |
|  |  |  |
| /\* Audio Data \*/ |  |  |
| if (audio\_signals\_in\_source\_block[block\_id] > 0) { |  |  |
| iif\_audio\_payload(); |  |  |
| } |  |  |
|  |  |  |
| byte\_alignment(); |  |  |
| } |  |  |

**Table 4.91 – Syntax of iif\_audio\_payload()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_audio\_payload() |  |  |
| { |  |  |
| b\_more\_elements = 1; |  |  |
| while (b\_more\_elements) |  |  |
| { |  |  |
| **b\_more\_elements;** | **1** | **uimsbf** |
| **b\_element\_is\_cpe;** | **1** | **uimsbf** |
| if (b\_element\_is\_cpe == 0) |  |  |
| { |  |  |
| single\_channel\_element(); |  |  |
| } |  |  |
| else |  |  |
| { |  |  |
| channel\_pair\_element(); |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 4.90 – Syntax of iif\_metadata\_payload()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_metadata\_payload(num\_block\_bits\_left) |  |  |
| { |  |  |
| num\_bits\_left = num\_block\_bits\_left; |  |  |
| num\_metadata\_sections = 0; |  |  |
| metadata\_section\_entry[num\_metadata\_sections] = -1; |  |  |
|  |  |  |
| if (use\_explicit\_metadata\_signaling[block\_id]) { |  |  |
| **b\_more\_md\_sections**; | **1** | **uimsbf** |
| while (b\_more\_md\_sections) { |  |  |
| **md\_type\_idx;** | **4** | **uimsbf** |
| metadata\_section\_entry[num\_metadata\_sections] =  md\_entry[block\_id][md\_type\_idx]; |  |  |
| num\_metadata\_sections += 1; |  |  |
| metadata\_section\_entry[num\_metadata\_sections] = -1; |  |  |
| **b\_more\_md\_sections;** | **1** | **uimsbf** |
| } |  |  |
| } |  |  |
| else { |  |  |
| for (m=0; m<num\_metadata\_configs[block\_id]; m++) { |  |  |
| **b\_metadata\_present;** | **1** | **uimsbf** |
| if (b\_metadata\_present) { |  |  |
| metadata\_section\_entry[num\_metadata\_sections] =  md\_entry[block\_id][m]; |  |  |
| num\_metadata\_sections += 1; |  |  |
| metadata\_section\_entry[num\_metadata\_sections] = -1; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
|  |  |  |
| if (num\_metadata\_sections > 0) { |  |  |
| for (s=0; s<num\_metadata\_sections; s++) { |  |  |
| num\_bits\_read = metadata\_section() |  |  |
| num\_bits\_left -= num\_bits\_read; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 4.91 – Syntax of metadata\_section()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| metadata\_section(num\_bits, byte\_aligned) |  |  |
| { |  |  |
| if (framing) { |  |  |
| **num\_metadata\_chunks\_in\_section\_minus1;** | **max\_chunks\_bits** | **uimsbf** |
| **b\_audio\_time\_code\_exists;** | **1** | **uimsbf** |
| if (b\_audio\_time\_code\_exists) { |  |  |
| **b\_audio\_time\_code\_format\_utc\_tai;** | **1** | **uimsbf** |
| **audio\_timestamp\_nanosecs;** | **64** | **uimsbf** |
| } |  |  |
| **b\_original\_audio\_time\_code\_exists;** | **1** | **uimsbf** |
| if (b\_original\_audio\_time\_code\_exists) { |  |  |
| **b\_original\_audio\_time\_code\_format\_utc\_tai;** | **1** | **uimsbf** |
| **original\_audio\_timestamp\_nanosecs;** | **64** | **uimsbf** |
| } |  |  |
|  |  |  |
| for (i = 0; i < num\_metadata\_chunks\_in\_section\_minus1 + 1; ++i) { |  |  |
| **chunk\_number;** | **max\_spreading\_bits** | **uimsbf** |
| if (chunk\_number == 0) { |  |  |
| **sample\_offset\_from\_audio;** | **12** | **uimsbf** |
| **num\_chunks\_in\_metadata\_frame\_minus1;** | **max\_spreading\_bits** | **uimsbf** |
| } |  |  |
|  |  |  |
| chunk\_len\_bytes = md\_size(is\_bytes=true) >> 3; |  |  |
| for (i = 0; i < chunk\_len\_bytes; ++i) { |  |  |
| **chunk\_byte;** | **8** | **uimsbf** |
| chunk\_buffer[md\_entry].push\_back(chunk\_byte) |  |  |
| } |  |  |
| if (chunk\_number == num\_chunks\_in\_metadata\_frame\_minus1) { |  |  |
| iif\_metadata\_frame(metadata\_type, chunk\_buffer.size()) |  |  |
| } |  |  |
| } |  |  |
| } else { |  |  |
| n\_metadata\_bits = md\_size(byte\_aligned); |  |  |
| iif\_metadata\_frame(metadata\_type, n\_metadata\_bits); |  |  |
| } |  |  |
| return n\_metadata\_bits; |  |  |
| } |  |  |

Note: chunk\_buffer.push\_back() inserts one byte at the end of the buffer

Note 2: chunk\_buffer.size() returns the size of the buffer in bytes

**Table 4.91 – Syntax of md\_size()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| md\_size(is\_bytes) |  |  |
| { |  |  |
| md\_size\_num\_bits\_base = 10 |  |  |
| md\_size\_num\_bits\_ext = 18 |  |  |
|  |  |  |
| if (is\_bytes) { |  |  |
| md\_size\_num\_bits\_base = 7 |  |  |
| md\_size\_num\_bits\_ext = 15 |  |  |
| } |  |  |
|  |  |  |
| **md\_size\_value;** | **md\_size\_num\_bits\_base** | **uimsbf** |
| if (md\_size\_value == 0) { |  |  |
| **ext\_md\_size\_value;** | **md\_size\_num\_bits\_ext** | **uimsbf** |
| md\_size\_value = (1 << md\_size\_num\_bits\_base) + ext\_md\_size\_value; |  |  |
| } |  |  |
| if (is\_bytes) { |  |  |
| md\_size\_value = md\_size\_value \* 8; |  |  |
| } |  |  |
| return md\_size\_value; |  |  |
| } |  |  |

**Table 4.91 – Syntax of iif\_metadata\_frame()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_metadata\_frame(\_md\_type, \_nbits) |  |  |
| { |  |  |
| switch (\_md\_type) { |  |  |
| case METADATA\_TYPE\_AUTH:  auth\_extension(); |  |  |
| break; |  |  |
| default: |  |  |
| **unknown\_metadata;** | **\_nbits** | **uimsbf** |
| break; |  |  |
| } |  |  |
|  |  |  |
| } |  |  |

**Table 4.91 – Syntax of iif\_config\_update()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_config\_update(\_cfg) |  |  |
| { |  |  |
| if (audio\_presentation\_type ==  AUDIO\_PRESENTATION\_TYPE\_CHANNELS) { |  |  |
| **active\_channel\_config\_idx[\_cfg];** | **3** | **uimsbf** |
| active\_channel\_mode[\_cfg] =  channel\_config\_set\_channel\_mode[active\_channel\_config\_idx[\_cfg]]; |  |  |
| active\_channel\_mask[\_cfg] =  channel\_config\_set\_channel\_mask[active\_channel\_config\_idx[\_cfg]]; |  |  |
|  |  |  |
| **b\_active\_channel\_config\_change[\_cfg];** | **1** | **uimsbf** |
| if (b\_active\_channel\_config\_change[\_cfg]) { |  |  |
| **next\_channel\_config\_sample\_offset\_minus1;** | **10** | **uimsbf** |
| next\_channel\_config\_sample\_offset[\_cfg] =  next\_channel\_config\_sample\_offset\_minus1 + 1; |  |  |
| **next\_channel\_config\_idx[\_cfg];** | **3** | **uimsbf** |
| next\_channel\_mode[\_cfg] =  channel\_config\_set\_channel\_mode  [next\_channel\_config\_idx[\_cfg]]; |  |  |
| next\_channel\_mask[\_cfg] =  channel\_config\_set\_channel\_mask  [next\_channel\_config\_idx[\_cfg]]; |  |  |
| } |  |  |
| } |  |  |
|  |  |  |
| if (num\_target\_devices > 0) { |  |  |
| /\* init with defaults \*/ |  |  |
| for (d=0; d<num\_target\_devices; d++) { |  |  |
| for (s=0; s<num\_signals\_for\_device\_routing; s++) { |  |  |
| audio\_signal\_role[\_cfg][d][s] = default\_audio\_signal\_role[d][s]; |  |  |
| audio\_signal\_gain[\_cfg][d][s] = default\_audio\_signal\_gain[d][s]; |  |  |
| audio\_signal\_delay[\_cfg][d][s] =  default\_audio\_signal\_delay[d][s]; |  |  |
| } |  |  |
| } |  |  |
| /\* override \*/ |  |  |
| **b\_override\_defaults;** | **1** | **uimsbf** |
| if (b\_override\_defaults == 1) { |  |  |
| for (d=0; d<num\_target\_devices; d++) { |  |  |
| if (device\_address\_mask[block\_id]&(1<<d)) { |  |  |
| /\* only if parent block is addressed to this device! \*/ |  |  |
| **b\_override\_default\_role\_for\_device;** | **1** | **uimsbf** |
| if (b\_override\_default\_role\_for\_device) { |  |  |
| for (s=0; s<num\_signals\_for\_device\_routing; s++) { |  |  |
| **b\_signal\_connected;** | **1** | **uimsbf** |
| if (b\_signal\_connected == 1) { |  |  |
| **audio\_signal\_role[\_cfg][d][s];** | **3** | **uimsbf** |
| } |  |  |
| else { |  |  |
| audio\_signal\_role[\_cfg][d][s] = -1; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| **b\_override\_default\_params\_for\_device;** | **1** | **uimsbf** |
| if (b\_override\_default\_params\_for\_device) { |  |  |
| for (s=0; s<num\_signals\_for\_device\_routing; s++) { |  |  |
| if (audio\_signal\_role[\_cfg][d][s] >= 0){ |  |  |
| **b\_override\_default\_gain;** | **1** | **uimsbf** |
| if (b\_override\_default\_gain) { |  |  |
| **gain\_code;** | **5** | **uimsbf** |
| **b\_override\_default\_delay;** | **1** | **uimsbf** |
| if (b\_override\_default\_delay) { |  |  |
| **b\_relative\_delay;** | **1** | **uimsbf** |
| if (b\_relative\_delay == 1) { |  |  |
| **rel\_delay\_code;** | **7** | **uimsbf** |
| } |  |  |
| else { |  |  |
| **delay\_code;** | **12** | **uimsbf** |
| audio\_signal\_delay[\_cfg][d][s] =  4\*delay\_code; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 4.91 – Syntax of iif\_block\_footer()**

|  |  |  |
| --- | --- | --- |
| Syntax | No. of bits | Mnemonic |
| iif\_block\_footer() |  |  |
| { |  |  |
| **block\_fill;** | **block\_bytes\_left \* 8** |  |
| if (b\_block\_protected) { |  |  |
| **block\_crc;** | **16** | **uimsbf** |
| } |  |  |
| } |  |  |

At the end of 4.5.2.9.1, add:

**extension\_type\_add** Four bit field indicating the value to be added to extension\_type to identify the type of fill element content according to Table 4.127.

Replace Table 4.127 with the following:

Table 4.127 – Values of the extension\_type field

|  |  |  |
| --- | --- | --- |
| Symbol | Value of extension\_type | Purpose |
| EXT\_FILL | ‘0000’ | bitstream payload filler |
| EXT\_FILL\_DATA | ‘0001’ | bitstream payload data as filler |
| EXT\_DATA\_ELEMENT | ’0010‘ | data element |
| EXT\_DATA\_LENGTH | ‘0011’ | container with explicit length for extension\_payload() |
| EXT\_UNI\_DRC | ’0100‘ | Unified dynamic range control |
| EXT\_AUTH\_DATA | ’0101‘ | Authentication Data |
| - | ’0110‘ | reserved |
| - | ’0111‘ | reserved |
| EXT\_ADD\_TYPES | ‘1000’ | Allows signalling of additional extension\_type values |
| EXT\_LDSAC\_DATA | ‘1001’ | LD MPEG Surround |
| EXT\_SAOC\_DATA | ‘1010’ | SAOC |
| EXT\_DYNAMIC\_RANGE | ‘1011’ | dynamic range control |
| EXT\_SAC\_DATA | ‘1100’ | MPEG Surround |
| EXT\_SBR\_DATA | ‘1101’ | SBR enhancement |
| EXT\_SBR\_DATA\_CRC | ‘1110’ | SBR enhancement with CRC |
| EXT\_SAOC\_DE\_DATA | ‘1111’ | SAOC-DE |
| Note: Extension payloads of the type EXT\_FILL or EXT\_FILL\_DATA have to be added to the bitstream payload if the total bits for all audio data together with all additional data are lower than the minimum allowed number of bits in this frame necessary to reach the target bitrate. Those extension payloads are avoided under normal conditions and free bits are used to fill up the bit reservoir. Those extension payloads are written only if the bit reservoir is full.  ‘reserved’ values can be used for a further extension of the syntax in a compatible way. | | |

After 4.5.2.16, add:

**4.5.2.17 Media Authentication**

MPEG-4 audio supports media authentication. The corresponding data is carried in an extension payload with the type EXT\_AUTH\_DATA. The syntax element auth\_extension() shall be used to embed media authentication information, as defined in Table 4.XXX.

**4.5.2.17.1 Media Authentication Syntax**

**Table 4.XXX — Syntax of auth\_extension**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| auth\_extension() |  |  |
| { |  |  |
| **numAuthExtMinus1** = escapedValue(4,4,4); | **4, 8, 12** | **uimsbf** |
| for (i=0; i <= numAuthExtMinus1; i++) { |  |  |
| **authSequence** | **1** | **uimsbf** |
| **authExtType** | **3** | **uimsbf** |
| switch (authExtType) { |  |  |
| case AUTH\_START: |  |  |
| authExtConfigAAC(); |  |  |
| break; |  |  |
| case AUTH\_SIG: |  |  |
| authExtSig(); |  |  |
| break; |  |  |
| case AUTH\_UUID: |  |  |
| authUUID(); |  |  |
| break; |  |  |
| case AUTH\_TIME: |  |  |
| authTimestamp(); |  |  |
| break; |  |  |
| default: |  |  |
| break; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 4.XXY — Syntax of authExtConfigAAC**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| authExtConfigAAC() |  |  |
| { |  |  |
| **authID** | **8** | **uimsbf** |
| **authHashType** = escapedValue(4,8,8); | **4, 12, 20** | **uimsbf** |
| **authKeyID** = escapedValue(3,8,8); | **3, 11, 19** | **uimsbf** |
| **authProvID** = escapedValue(8,8,16); | **8, 16, 32** | **uimsbf** |
| if (authProvID == 0x00) { |  |  |
| **authSourceURILengthMinus1** | **8** | **uimsbf** |
| for (i=0; i<= authProvIDLengthMinus1; i++) { |  |  |
| **authSourceURI[i]** | **8** | **Bslbf** |
| } |  |  |
| } |  |  |
|  |  |  |
| **isAuthCRC** | **1** | **uimsbf** |
|  |  |  |
| if((audioObjectType != 47){ //not AAC-IIF |  |  |
| **authAddExtTypes** | **1** | **uimsbf** |
| if (authAddExtTypes){ |  |  |
| **authAddExtInclusion** | **1** | **uimsbf** |
| **authAddExtTypeListLengthMinus1** | **3** | **uimsbf** |
| for (i=0; i <= authExtTypeListLengthMinus1; i++) { |  |  |
| **authAddExtType[i]** | **4** | **uimsbf** |
| } |  |  |
| } |  |  |
| } else { |  |  |
| **authDeviceSpecific** | **1** | **uimsbf** |
| if(authDeviceSpecific){ |  |  |
| **auth\_device\_mask** | **num\_target\_devices** | **uimsbf** |
| } |  |  |
|  |  |  |
| **authAllBlockStreams** | **1** | **uimsbf** |
| if (!authAllBlockStreams){ |  |  |
| **authAddBlockExtInclusion** | **1** | **uimsbf** |
| **authAddBlockExtTypeListLengthMinus1** | **3** | **uimsbf** |
| for (i=0;i<=authAddBlockExtTypeListLengthMinus1;i++) { |  |  |
| **authBlockID[i]** | **5** | **uimsbf** |
| } |  |  |
| } |  |  |
|  |  |  |
| **authAllPriorities** | **1** | **uimsbf** |
| if(!authAllPriorities){ |  |  |
| **authBlockPrioExtInclusion** | **1** | **uimsbf** |
| **authBlockPrioListLengthMinus1** | **2** | **uimsbf** |
| for (i=0; i<=authBlockPrioListLengthMinus1; i++) { |  |  |
| **authBlockPriority[i]** | **2** | **uimsbf** |
| } |  |  |
| } |  |  |
|  |  |  |
| **authAllMetadata** | **1** | **uimsbf** |
| if (!authAllMetadata){ |  |  |
| **authMetadataExtInclusion** | **1** | **uimsbf** |
| **authMetadataTypeListLengthMinus1** | **3** | **uimsbf** |
| for (i=0; i<=authMetadataTypeListLengthMinus1; i++) { |  |  |
| authMetadataType [i] = metadata\_type; | **8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table XXY — Syntax of authExtSig**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| authExtSig() |  |  |
| { |  |  |
| **authID** | **8** | **uimsbf** |
| **authPartialSig** | **1** | **uimsbf** |
| **authABREnable** | **2** | **uimsbf** |
| numSig = 1; |  |  |
|  |  |  |
| if (authABREnable) { |  |  |
| **authABRBitrateAtMinMax** | **2** | **uimsbf** |
|  |  |  |
| if (authABREnable==1) { |  |  |
| if(authABRBitrateAtMinMax==1) { |  |  |
| numSig = 2; |  |  |
| } else { |  |  |
| numSig = 3; |  |  |
| } |  |  |
| } |  |  |
| if (authABREnable==2) { |  |  |
| if(authABRBitrateAtMinMax==1) { |  |  |
| numSig = 3; |  |  |
| } else if(authABRBitrateAtMinMax==2) { |  |  |
| numSig = 4; |  |  |
| } else { |  |  |
| numSig = 5; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| for (n=0; n<numSig; n++) { |  |  |
| if (authPartialSig) { |  |  |
| **sigSegmentStart** | **1** | **uimsbf** |
| **sigSegmentStop** | **1** | **uimsbf** |
| **sigSegmentLengthMinus1** | **5** | **uimsbf** |
| **sigPartial** | **(sigSegmentLengthMinus1+1)\*8** | **uimsbf** |
| } else { |  |  |
| **sigLengthMinus1** | **7** | **uimsbf** |
| **sigComplete** | **(sigLengthMinus1+1)\*8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 3 — Syntax of authUUID**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| authUUID() |  |  |
| { |  |  |
| **uuidSegmentStart** | **1** | **uimsbf** |
| **uuidSegmentStop** | **1** | **uimsbf** |
| **uuidSegmentLengthMinus1** | **4** | **uimsbf** |
| **uuid** | **(uuidSegmentLengthMinus1+1)\*8** | **uimsbf** |
| } |  |  |

**Table 4 — Syntax of authTimestamp**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| authTimestamp() |  |  |
| { |  |  |
| **authID** | **8** | **uimsbf** |
| **authTimeType** | **7** | **uimsbf** |
| **authTimeOffsetType** | **1** | **uimsbf** |
| switch (authTimeType ) { |  |  |
| case authTimeLong: |  |  |
| **authTime** = escapedValue(28,32,0); | **28, 60** | **uimsbf** |
| **authTimeOffset** | **12** | **uimsbf** |
| break; |  |  |
| case authTimeShort: |  |  |
| **authTimeS** = escapedValue(4,8,8); | **4, 12, 20** | **uimsbf** |
| **authTimeOffsetS** = escapedValue(4,8,8); | **4, 12, 20** | **uimsbf** |
| break; |  |  |
| case authTimeTAI: { /\* acc. ISO/IEC 23001-17 \*/ |  |  |
| **TAI\_timestamp;** | **64** | **uimsbf** |
| **status\_bits;** | **8** | **uimsbf** |
| break; |  |  |
| default: |  |  |
| break; |  |  |
| } |  |  |
| } |  |  |

**4.5.2.17.2 Media Authentication Semantics**

|  |  |
| --- | --- |
| **numAuthExtMinus1** | Plus 1 indicates the number of signalled authExtType elements. |
| **authSequence** | Indicates the authentication sequence to which the related authentication information belongs to. |
| **authExtType** | Indicates the type of authentication information signalled according to Table XYZ. |
| **Table XYZ — Value and meaning of authExtType**   |  |  | | --- | --- | | **authExtType** | **Value** | | AUTH\_START | 0 | | AUTH\_SIG | 1 | | AUTH\_UUID | 2 | | AUTH\_TIME | 3 | | Reserved for ISO use | 4-7 | | |
|  | |
| **authID** | Shall be used to identify the combination of authHashType, authProvID and authKeyID to which the related authentication information belongs to. This may be used to enable authentication of one authentication sequence with different authentication configurations. |
| **authHashType** | Indicates the hashing algorithm used according to Table XYZ. |
| **Table XYZ — Value and meaning of authHashType**   |  |  | | --- | --- | | **authHashType Value** | **Hashing algorithm** | | 0 | SHA-1 | | 1 | SHA-224 | | 2 | SHA-256 | | 3 | SHA-384 | | 4 | SHA-512 | | All other values | /\* reserved for ISO use \*/ | | |
|  | |
| **escapedValue()** | See ISO/IEC 23003-3. |
| **authKeyID** | Identifies the authentication key used to calculate the value of the signature in authExtSig(). The values of authKeyID are dependent on the authentication provider and are not defined in the present document. This value shall be set to 0 in case there is no key needed for the underlying hashing function. |
| **authAddExtTypes** | Indicates if authentication information for additional extension types is signalled. |
| **authProvID** | Identifies the provider of the authentication system according to Table XYZ. In case authProvID equals to 1, there is no provider. This mode can be used to create a message digest only by using the method identified by authHashType. |
| **Table XYZ — Value and meaning of authProvID**   |  |  | | --- | --- | | authProvID Value | Authentication Provider URI | | 0 | See authSourceURI. | | 1 | Message Digest only | | 2-… | /\* Registration Authority \*/ | | |
| **authSourceURILengthMinus1** | Plus 1 indicates the length of the authSourceURI-field in bytes. |
| **authSourceURI** | Contains a URI with syntax and semantics as defined in as defined in IETF RFC 3986. |
| **isAuthCRC** | Indicates if an CRC-based authentication method is used. If set to 1, the used verification mechanism shall generate the signature (sigPartial or sigComplete) utilizing the data resulting from calculating a CRC as defined in 1.8.4.5 (CRC16), using the syntax elements according to 4.5.2.17.3 as input. |
| **authAddExtInclusion** | If set to 1, all extension types signalled in authAddExtType shall be included into the calculation of the authentication information, according to 4.5.2.17.3. If set to 0, all extension types signalled in authAddExtType shall be excluded from the calculation of the authentication information, according to 4.5.2.17.3. |
| **authAddExtTypeListLengthMinus1** | Plus 1 indicates the length of the list of authAddExtType. |
| **authAddExtType** | Indicates the extension type to be included or excluded for the calculation of the authentication information. |
| **authDeviceSpecific** | Indicates that device specific authentication information is available. |
| **auth\_device\_mask** | For a given authID, specifies a device mask indicating devices that shall provide (through an interface) or use the provided authentication information to allow for verifying the authentication information. Bit 0 of this mask refers to the least significant bit of that mask and is associated with the device\_id=’0’, bit 1 with device\_id=’1’, and other bits respectively. |
| **authAllBlockStreams** | Indicates if all IIF block streams shall be included for the calculation of the authentication information. |
| **authAddBlockExtInclusion** | If set to 1, all IIF block streams corresponding to the ID signalled in authBlockID shall be included into the calculation of the authentication information, according to 4.5.2.17.3. If set to 0, all IIF block streams corresponding to the ID signalled in authBlockID shall be excluded from the calculation of the authentication information, according to 4.5.2.17.3. |
| **authAddBlockExtTypeListLengthMinus1** | Plus 1 indicates the length of the list of authBlockID. |
| **authBlockID** | Indicates the block\_id of the IIF block stream to be included or excluded for the calculation of the authentication information. |
| **authAllPriorities** | Indicates if all IIF Blocks shall be included for the calculation of the authentication information, independent of their respective block\_priority. |
| **authBlockPrioExtInclusion** | If set to 1, all IIF Blocks with the same block\_priority as signalled in authBlockPriority shall be included into the calculation of the authentication information, according to 4.5.2.17.3. If set to 0, all IIF Blocks with the same block\_priority as signalled in authBlockPriority shall be excluded from the calculation of the authentication information, according to 4.5.2.17.3. |
| **authBlockPrioListLengthMinus1** | Plus 1 indicates the length of the list of authBlockPriority. |
| **authBlockPriority** | Indicates the block\_priority of the IIF Block to be included or excluded for the calculation of the authentication information. |
| **authAllMetadata** | Indicates if all iif\_metadata\_frame() elements shall be included for the calculation of the authentication information, independent of their respective metadata\_type. |
| **authMetadataExtInclusion** | If set to 1, all iif\_metadata\_frame()-elements with the same metadata\_type as signalled in authMetadataType shall be included into the calculation of the authentication information, according to 4.5.2.17.3. If set to 0, all iif\_metadata\_frame()-elements with the same metadata\_type as signalled in authMetadataType shall be excluded from the calculation of the authentication information, according to 4.5.2.17.3. |
| **authMetadataTypeListLength**  **Minus1** | Plus 1 indicates the length of the list of authMetadataType. |
| **authMetadataType** | Indicates the metadata\_type of the iif\_metadata\_frame()-element to be included or excluded for the calculation of the authentication information. |
| **authABREnable** | Indicates the type of the adaptive bitrate (ABR) scheme and the respective number of signature values according to Table XYZ.  In the case that authABREnable equals 1 (upDownABR), the order of encapsulation is as follows: signatures for the streams at the current bitrate, the bitrate one above, the bitrate one below.  In the case that authABREnable equals 2 (upDownMinMaxABR), the order of encapsulation is as follows: signatures for the streams at the current bitrate, the bitrate one above, the bitrate one below, the maximum bitrate, the minimum bitrate. |
| **Table XYZ — Value and meaning of authABREnable**   |  |  | | --- | --- | | **authABREnable Value** | **ABR Scheme** | | 0 | noABR | | 1 | upDownABR | | 2 | upDownMinMaxABR | | 3 | /\* reserved \*/ | | |
|  | |
| **authABRBitrateAtMinMax** | Indicates whether the current bitrate fulfils one of the conditions according to Table XYZ, that are related to the minimum and maximum bitrates in the adaptation set. In either case, the number of signature values is reduced. |
| **Table XYZ — Value and meaning of authABRBitrateMinMax** | |
| |  |  | | --- | --- | | **authABRBitrateMinMax Value** | **ABR Bitrate Status** | | 0 | Default value | | 1 | The current bitrate is equal to the maximum or minimum bitrate in the adaptation set | | 2 | The current bitrate is one level below the maximum or one level above the minimum bitrate in the adaptation set | | 3 | /\* reserved \*/ | | |
|  |  |
| **authPartialSig** | If set to ‘1’, indicates that the signature is transmitted partially. |
| **sigSegmentStart** | Indicates that the following sigPartial is the first segment of a signature. |
| **sigSegmentStop** | Indicates that the following sigPartial is the last segment of a signature.  NOTE: If both sigSegmentStartand and sigSegmentStop are equal to ‘1’, sigPartial contains a signature which is complete, but shorter than a full signature resulting from the related hashing algorithm. |
| **sigSegmentLengthMinus1** | Plus 1 indicates the length of the sigPartial field. |
| **sigPartial** | This field carries a segment of the signature resulting from the used verification mechanism. sigPartial shall be calculated in the same way as sigComplete. sigPartial may be created by truncating sigComplete.  Note: Verification may happen comparing only a subset of the bits resulting from the hashing algorithm. |
| **sigLengthMinus1** | Plus 1 indicates the length of the sigComplete field in bytes. |
| **sigComplete** | This field carries the signature resulting from the used verification mechanism. sigComplete shall be calculated by applying the hashing algorithm and configuration as signalled in the authExtConfigAAC() syntax element. The input data to the hashing algorithm shall be created in alignment with 4.5.2.17.3.3.  Note: Verification may happen comparing only a subset of the bits resulting from the hashing algorithm. |
| **uuidSegmentStart** | If set to ‘1’, indicates if the bytes in the uuid field are the first bytes of a UUID-segment. |
| **uuidSegmentStop** | If set to ‘1’, indicates if the bytes in the uuid field are the last bytes of a UUID-segment.  Note: If both uuidSegmentStart and uuidSegmentStop are set to ‘1’ the uuid field contains the full UUID. |
| **uuidSegmentLengthMinus1** | Plus 1 indicates the length of the uuid-field in bytes. |
| **uuid** | This field contains the UUID of the related (sub-)stream. It may be used to map the audio stream to other media types, such as a related video stream. |
| **authTimeType** | Indicates the type of the time-signalling according to Table XYZ. |
| **Table XYZ — Value and meaning of authTimeType**   |  |  | | --- | --- | | **authTimeType Value** | **Timing Scheme** | | 0 | authTimeLong | | 1 | authTimeShort | | 2 | authTimeTAI | | 3-127 | /\* reserved \*/ | | |
|  | |
| **authTimeOffsetType** | Indicates the unit of *authTimeOffset* value. This shall be set to ‘0’ if the unit is milliseconds, and shall be set to ‘1’ for using the sampling rate configured for the underlying signal type as base time. |
| **authTime** | Indicates the base time. This is counted in seconds and the count starts on January 1st, 2025 at 00.00.01 UTC. |
| **authTimeOffset** | Indicates the time offset compared to the value indicated in authTime. The time shall be set in a way that authTime + authTimeOffset indicates the time when the first sample of following related access unit has been recorded. |
| **authTimeS** | Indicates the base time elapsed in seconds since the last time update of type ‘authTimeLong’. |
| **authTimeOffsetS** | Indicates the time offset compared to the value indicated in authTimeS. The time shall be set in a way that authTime + authTimeOffset + authTimeS + authTimeOffsetS indicates the time when the first sample of following related access unit has been recorded. |
| **TAI\_timestamp** | Indicates the TAI\_timestamp according to ISO/IEC 23001-17. |
| **status\_bits** | Indicates the bits synchronization\_state, timestamp\_generation\_failure, timestamp\_is\_modified and reserved according to ISO/IEC 23001-17. |

**4.5.2.17.3 Media Authentication Interface**

For applications which require media authentication, related data for verification of the authenticity of a bitstream and mapping to other media types shall be provided to the system by using the syntax element mpeg4audio\_GetAuthData().

**4.5.2.17.3.1 Syntax**

**Table 5 — Syntax of mpeg4audio\_GetAuthData**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| mpeg4audio\_GetAuthData() |  |  |
| { |  |  |
| authExtConfigAAC() |  |  |
| authExtSig() |  |  |
| authUUID() |  |  |
| authTimestamp() |  |  |
| **gad\_bytesLengthMinus1** | **64** | **uimsbf** |
| **gad\_bytes** | **(gad\_bytesLengthMinus1+1)\*8** | **uimsbf** |
| } |  |  |

**4.5.2.17.3.2 Semantics**

|  |  |
| --- | --- |
| **gad\_bytesLengthMinus1** | Plus 1 indicates the length of the gad\_bytes field. |
| **gad\_bytes** | Includes all bytes relevant for the creation of the authentication information according to the configuration described in authExtConfigAAC(). |

**4.5.2.17.3.3 Processing**

The decoder shall extract all data related to media authentication for every authentication sequence as indicated by authID and authSequence, and populate the respective fields in mpeg4audio\_GetAuthData().

authExtConfigAAC() and authExtSig() for the related authentication sequence (as indicated by the respective authID and authSequence) shall be copied from the respective bitstream elements. authTimestamp() having the same authID as the current authentication sequence and authUUID() syntax elements shall always be copied from the latest occurrences in bitstream. The gad\_bytes field shall be populated by concatenating all bytes of all access units starting from the access unit which includes the related authExtConfigAAC() syntax element up until and including the access unit which contains the related authExtSig() syntax element in which either sigSegmentStop equals ‘1’ or authPartialSig equals ‘0’. The authExtConfigAAC() syntax element of the respective authentication sequence shall be populated into gad\_bytes. The authExtSig() syntax element of the respective authentication sequence shall not be populated into gad\_bytes. If any of the access units related to the current authentication sequence contains authExtConfigAAC() or authExtSig() syntax elements with the same authID as the current authentication sequence and a different authSequence value, they shall be populated into gad\_bytes. authExtConfigAAC() and authExtSig() syntax elements with a different authID than the current authentication sequence and authUUID() elements shall not be populated into gad\_bytes. The following extension types indicated by extension\_type in Table 4.127 shall be excluded by default and not be populated into gad\_bytes:

* EXT\_FILL
* EXT\_FILL\_DATA
* EXT\_DATA\_ELEMENT
* EXT\_DATA\_LENGTH

All other extension types shall be included and populated into gad\_bytes, unless they are signalled to be excluded via authAddExtType and authAddExtInclusion equal to ‘0’. The extension types excluded by default may be signalled to be included using authAddExtInclusion equal to ‘1’ and setting the respective authAddExtType value.

For AAC IIF, in addition to the selection of data based on access units for gad\_bytes described above, additional rules shall be applied for the selection of data contained in iif\_access\_unit() syntax elements. iif\_block() and iif\_metadata\_frame() syntax elements shall be included or excluded depending on the semantics of the following fields:

* authAllBlockStreams, authAddBlockExtInclusion, authBlockID,
* authAllPriorities, authBlockPrioExtInclusion, authBlockPriority,
* authAllMetadata, authMetadataExtInclusion and authMetadataType.

In the case an extension type or syntax element is excluded, the respective bits shall be replaced with ‘0’s before population into gad\_bytes.

The input data for the hashing algorithm shall be created by concatenating gad\_bytes with the bytes contained in the uuid field, if existing.

**4.5.2.18 Payloads for the audio object type IIF**

**4.5.2.18.1 Semantics**

|  |  |
| --- | --- |
| **frame\_counter** | Indicates the IIF frame to which this IIF block belongs to. For all frame lengths except 200.2 and 400.4, the value shall be counted from 0 to 15 and wrap around to 0. For frame lengths 200.2 and 400.4 the value shall be counted from 0 to 14 and wrap around to 0. |
| **block\_id** | Indicates the current IIF block. The IIF Block ID links a block to the corresponding block stream. Values shall be strictly contiguous and in ascending order. |
| **block\_priority** | Indicates the priority of the IIF block, 0 being the highest priority. A value of ‘0’ shall only be used for the IIF blocks belonging to the current IIF frame in an access unit. There shall only be one IIF block of each block\_id with a block\_priority value of ‘0’ belonging to one IIF frame.  A value of ‘1’ shall be used to signal the first layer of redundant IIF blocks. There may be multiple IIF blocks with the same block\_id and a block\_priority value of ‘1’ belonging to one IIF frame.  A value of ‘2’ shall be used to signal the second layer of redundant IIF blocks. There may be multiple IIF blocks with the same block\_id and a block\_priority value of ‘2’ belonging to one IIF frame.  A value of ‘3’ shall be used to signal the third layer of redundant IIF blocks. There may be multiple IIF blocks with the same block\_id and a block\_priority value of ‘3’ belonging to one IIF frame. |
| **block\_size** | Indicates the size of the IIF block in bytes, including the first element after the reserved element and all elements of iif\_block\_footer() element. The total size of a block is therefore block\_size+3 bytes. |
| **b\_block\_protected** | Indicates if the block is protected by a CRC. The value shall be set to ‘0’ in the case of a CRC not being present. The value shall be set to ‘1’ in the case of a CRC being present. |
| **block\_size\_extension** | Indicates the extension size of the IIF block in bytes. |
| **b\_more\_bits** | Indicates if there’s further block size extension in the bitstream. |
| **b\_config\_update\_inband** | Indicates if the config update is included in-band. |
| **dynamic\_config\_upd\_idx** | Indicates the index of the configuration update. A value of ‘0’ is reserved for inband configuration updates and shall not to be used. |
| **md\_type\_idx** | Indicates the index of the metadata type. |
| **b\_more\_md\_sections** | Indicates if there are more sections of metadata present. |
| **b\_metadata\_present** | Indicates if metadata are present. |
| **num\_metadata\_chunks\_in\_section\_minus1** | Indicated the number of metadata chunks of the indicated type in that metadata section minus 1.. |
| **b\_audio\_time\_code\_exists** | Indicates the presence of an audio timestamp. The audio timestamp shall represent the creation time of the first sample in the audio payload within the context of the product.. |
| **b\_audio\_time\_code\_format\_utc\_tai** | Indicates the time system of the audio timestamp. 0: UTC, 1: TAI. |
| **audio\_timestamp\_nanosecs** | Indicates the unit of the value is nanoseconds from the start of the Epoch (1st Jan 1970). The time system used is either UTC or TAI depending on the value of **b\_audio\_time\_code\_format\_utc\_tai**. |
| **b\_original\_audio\_time\_code\_exists** | Indicates the presence of an original audio timestamp. The original audio timestamp shall represent the creation time of the first sample in the audio payload within the context of the entire workflow. This value is established when the audio is originally created, and the accompanying video is also recorded. |
| **b\_original\_audio\_time\_code\_format\_utc\_tai** | Indicates the time system of the original audio timestamp. 0: UTC, 1: TAI. |
| **original\_audio\_timestamp\_nanosecs** | Indicates the unit of the value is nanoseconds from the start of the Epoch (1st Jan 1970). The time system used is either UTC or TAI depending on the value of **b\_original\_audio\_time\_code\_format\_utc\_tai**. |
| **chunk\_number** | Indicates the current metadata chunk number, counting from 0 to num\_chunks\_in\_metadata\_frame\_minus1 strictly continuously. The counting can be done across blocks of the same frame and across frames. |
| **sample\_offset\_from\_audio** | Indicates the sample offset of the metadata into the frame that contains the last metadata chunk. Value 0 means the frame boundary, a value of 1 means that the metadata starts after the first sample and so on. |
| **num\_chunks\_in\_metadata\_frame\_minus1** | Indicates the number of chunks the current metadata frame is spread out to minus 1. |
| **chunk\_byte** | Indicates one byte of a metadata chunk. When all bytes of all related chunks are available, they shall be interpreted as iif\_metadata\_frame() bitstream element. |
| **b\_more\_elements** | Indicates continuation of the related while-loop. |
| **b\_element\_is\_cpe** | Indicates if the element is a CPE or SCE. |
| **md\_size\_value** | Indicates the size of the metadata. |
| **ext\_md\_size\_value** | Indicates the size-extension of the metadata. |
| **unknown\_metadata** | Contains bits for unknown types of metadata. |
| **active\_channel\_config\_idx** | Indicates the active channel configuration for the current IIF frame as index into the channel\_config\_set\_channel\_mode and channel\_config\_set\_channel\_mask array. |
| **b\_active\_channel\_config\_change** | Indicates if there is a change of the active channel configuration present in the current IIF frame.  A value of ‘0’ indicates that there is no channel configuration change in the current IIF frame. The channel configuration set by active\_channel\_config\_idx shall be valid for the duration of the entire frame.  A value of ‘1’ indicates that there is a channel config change within the current IIF frame. |
| **next\_channel\_config\_sample\_offset\_minus1** | Indicates after which sample, counting from the first sample of the associated IIF Block, the new channel configuration identified by the next\_channel\_config\_idx field shall be active. |
| **next\_channel\_config\_idx** | Indicates the active channel configuration for the current IIF frame, that shall be valid after the configuration change. |
| **b\_override\_defaults** | Indicates that all default values of the device specific routing are overridden for this IIF frame. |
| **b\_signal\_connected** | Indicates if a signal is connected and a role is assigned to a specific device. |
| **audio\_signal\_role** | Indicates the role of the audio signal according to Table 4.ASR. |
| **Table 4.ASR – Values of the audio\_signal\_role field**   |  |  |  | | --- | --- | --- | | Symbol | Value of audio\_signal\_role | Purpose | | AUDIO\_SIGNAL\_ROLE\_MAIN | ‘000’ | Main audio. | | AUDIO\_SIGNAL\_ROLE\_AUX1 | ‘001’ | Auxiliary audio. | | AUDIO\_SIGNAL\_ROLE\_AUX2 | ‘010’ | Auxiliary audio. | | AUDIO\_SIGNAL\_ROLE\_AUX3 | ‘011’ | Auxiliary audio. | | AUDIO\_SIGNAL\_ROLE\_AUX4 | ‘100’ | Auxiliary audio. | | AUDIO\_SIGNAL\_ROLE\_AUX5 | ‘101’ | Auxiliary audio. | | AUDIO\_SIGNAL\_ROLE\_AUX6 | ‘110’ | Auxiliary audio. | | AUDIO\_SIGNAL\_ROLE\_AUX7 | ‘111’ | Auxiliary audio. | | |
|  |  |
| **b\_override\_default\_params\_for\_device** | Indicates if gain- or delay-values are signaled for the related signal and device in this IIF frame.  A value of ‘0’ indicates that all default gain- and delay-values shall be applied for the related device in this IIF frame.  A value of ‘1’ indicates that gain- and/or delay-values are signaled. |
| **b\_override\_default\_gain** | Indicates if a gain is signaled for the related signal and device in this IIF frame.  A value of ‘0’ indicates that the default gain value shall be applied for the related device in this IIF frame.  A value of ‘1’ indicates that a gain value is signaled. |
| **gain\_code** | Indicates the gain code to derive the related audio signal gain. The audio signal gain value shall be calculated as follows: |
| **b\_override\_default\_delay** | Indicates if the default gain shall be applied to the audio signal s on device d.  A value of ‘0’ indicates that the default delay shall be applied.  A value of ‘1’ indicates that a delay is signaled for the audio signal s on the device d. |
| **b\_relative\_delay** | Indicates if the signaled delay is relative to the default delay.  A value of ‘0’ indicates that a new absolute delay is signaled that shall replace the default delay.  A value of ‘1’ indicates that a relative delay is signaled which shall be added to the default delay to determine the delay to be applied to the current IIF frame. |
| **rel\_delay\_code** | Indicates the relative delay code. The relative delay in samples shall be calculated as follows:  The delay in samples of the signal for device shall be calculated as follows: |
| **delay\_code** | Indicates the absolute delay code. The delay in samples of the signal s for device d shall be calculated as follows: |
| **block\_fill** | Indicates fill bits which shall be ignored. |
| **block\_crc** | Indicates a 16-bit CRC which shall be calculated over the concatenated data of iif\_block\_header(), iif\_block\_payload(), and the block\_fill element, as defined in 1.8.4.5 (CRC16). |

In 4.5.4 Tables add before Table 4.166 – AAC error sensitivity category assignment for main payload:

**Table 4.150 – scalefactor bands for a window length of 512 for LONG\_WINDOW**

|  |  |
| --- | --- |
| num\_swb\_long\_window | 20 |
| swb | swb\_offset\_lon  g\_window |
| 0 | 0 |
| 1 | 4 |
| 2 | 8 |
| 3 | 12 |
| 4 | 16 |
| 5 | 20 |
| 6 | 24 |
| 7 | 28 |
| 8 | 36 |
| 9 | 44 |
| 10 | 52 |
| 11 | 64 |
| 12 | 76 |
| 13 | 92 |
| 14 | 108 |
| 15 | 128 |
| 16 | 148 |
| 17 | 172 |
| 18 | 196 |
| 19 | 244 |
|  | 256 |

**Table 4.150 – scalefactor bands for a window length of 480 for LONG\_WINDOW**

|  |  |
| --- | --- |
| num\_swb\_long\_window | 20 |
| swb | swb\_offset\_lon  g\_window |
| 0 | 0 |
| 1 | 4 |
| 2 | 8 |
| 3 | 12 |
| 4 | 16 |
| 5 | 20 |
| 6 | 24 |
| 7 | 28 |
| 8 | 32 |
| 9 | 40 |
| 10 | 48 |
| 11 | 56 |
| 12 | 68 |
| 13 | 80 |
| 14 | 96 |
| 15 | 112 |
| 16 | 132 |
| 17 | 156 |
| 18 | 180 |
| 19 | 208 |
|  | 240 |

**Table 4.xx – scalefactor bands for a window length of 400 and 401 for LONG\_WINDOW**

|  |  |
| --- | --- |
| num\_swb\_long\_window | 19 |
| swb | swb\_offset\_lon  g\_window |
| 0 | 0 |
| 1 | 4 |
| 2 | 8 |
| 3 | 12 |
| 4 | 16 |
| 5 | 20 |
| 6 | 24 |
| 7 | 28 |
| 8 | 32 |
| 9 | 40 |
| 10 | 48 |
| 11 | 56 |
| 12 | 68 |
| 13 | 80 |
| 14 | 96 |
| 15 | 112 |
| 16 | 132 |
| 17 | 152 |
| 18 | 176 |
|  | 200/201 |

**Table 4.xx – scalefactor bands for a window length of 500 and 501 for LONG\_WINDOW**

|  |  |
| --- | --- |
| num\_swb\_long\_window | 20 |
| swb | swb\_offset\_lon  g\_window |
| 0 | 0 |
| 1 | 4 |
| 2 | 8 |
| 3 | 12 |
| 4 | 16 |
| 5 | 20 |
| 6 | 24 |
| 7 | 28 |
| 8 | 36 |
| 9 | 44 |
| 10 | 52 |
| 11 | 64 |
| 12 | 76 |
| 13 | 92 |
| 14 | 108 |
| 15 | 128 |
| 16 | 148 |
| 17 | 172 |
| 18 | 196 |
| 19 | 224 |
|  | 250/251 |

**Table 4.xx – scalefactor bands for a window length of 800 and 801 for LONG\_WINDOW**

|  |  |
| --- | --- |
| num\_swb\_long\_window | 33 |
| swb | swb\_offset\_lon  g\_window |
| 0 | 0 |
| 1 | 4 |
| 2 | 8 |
| 3 | 12 |
| 4 | 16 |
| 5 | 20 |
| 6 | 24 |
| 7 | 28 |
| 8 | 32 |
| 9 | 36 |
| 10 | 40 |
| 11 | 44 |
| 12 | 48 |
| 13 | 52 |
| 14 | 56 |
| 15 | 64 |
| 16 | 72 |
| 17 | 80 |
| 18 | 88 |
| 19 | 96 |
| 20 | 108 |
| 21 | 120 |
| 22 | 132 |
| 23 | 144 |
| 24 | 156 |
| 25 | 172 |
| 26 | 188 |
| 27 | 212 |
| 28 | 240 |
| 29 | 272 |
| 30 | 304 |
| 31 | 336 |
| 32 | 368 |
|  | 400/401 |
|  |  |

For the AAC Immersive Interchange Format, the scalefactor bands of Table 4.148 are reused for LONG\_WINDOW (256/240 window size).

After 4.6.20, add the following:

* + 1. ***AAC Immersive Interchange Format***
       1. **Introduction**

The AAC Immersive Interchange Format is an audio codec that enables reliable immersive audio transmission via transport links at low latency. It enables carriage of different types of metadata, which can be associated with the audio data. Further, it includes signaling which allows flexible handling of various use cases.

* + - 1. **Coder description**

The core codec of the AAC Immersive Interchange Format is defined by the following modifications with respect to the Low delay codec algorithm as described in 4.6.17.

* + - 1. **Frame length/window length**

In addition to 1024 and 960 windows (512 and 480 spectral values), the following window and frame lengths shall be supported:

* 2048/1920 resulting in 1024/960 spectral values
* 512/480 resulting in 256/240 spectral values
* 256/240 resulting in 128/120 spectral values
* 400/401 resulting in 200/201 spectral values
* 500/501 resulting in 250/251 spectral values
* 800/801 resulting in 400/401 spectral values
* 1000/1001 resulting in 500/501 spectral values

For odd window lengths N (401, 501, 801, 1001), the analytical expression of the IMDCT is:

Where:

|  |  |
| --- | --- |
| n | sample index |
| i | window index |
| k | spectral coefficient index |
| N | window length |
|  |  |

Fractional frame lengths shall be realized by combining frame lengths as described in Table 4.xxx:

**Table 4.xxx – sequences for fractional frame lengths**

|  |  |  |  |
| --- | --- | --- | --- |
| Frame length nominal | Sequence | Length of sequence | Range of frame\_counter |
| 200.2 | 200, 200, 200, 200, 201 | 5 | [0,14] |
| 250.25 | 250, 250, 250, 251 | 4 | [0,15] |
| 400.4 | 400, 400, 401, 400, 401 | 5 | [0,14] |
| 500.5 | 500, 501 | 2 | [0,15] |

The index into the sequence shall be obtained frame-by-frame as:

Where:

|  |  |
| --- | --- |
| c | value of frame\_counter of the current block |
|  | any sequence according to Table 4.xxx |
|  | length of the sequence |

For transitions from and to odd window lengths, the windowing and overlap-add is modified, compared to subclauses 4.6.11.3.2, 4.6.11.3.3, 4.6.17.2.3, in the following way.

Current frame is odd; previous frame was even, e.g., ; :

Current frame is even; previous frame was odd, e.g., ,

* + - 1. **Window shape**

The sine window (window\_shape==0) as defined in 4.6.11.3.2 and low-overlap window (window\_shape==1) as defined in 4.6.17.2.3 shall be reused. Additional values of N for the calculation of the sine window shall be 1024, 1000, 960, 800, 512, 500, 480 and 400. Additional values of N for the calculation of the low-overlap window shall be 2048, 1920, 1000, 800, 512, 500, 480, 400, 256 and 240.

For odd window lengths (401, 501, 801, 1001), the window sequence shall be derived from their even counterparts (400, 500, 800, 1000):

* + - 1. **Tables for temporal noise shaping (TNS)**

The following tables specify additional value of TNS\_MAX\_BANDS for the core codec of the AAC Immersive Interchange Format:

**Table xxx – TNS\_MAX\_BANDS in case of 128 and 120 samples per frame**

|  |  |
| --- | --- |
| Sampling rate | TNS\_MAX\_BANDS |
| 48000 | 14 |
| 44100 | 14 |

**Table xxx – TNS\_MAX\_BANDS in case of 256, 251, 250, 240, 201 and 200 samples per frame**

|  |  |
| --- | --- |
| Sampling rate | TNS\_MAX\_BANDS |
| 48000 | 17 |
| 44100 | 17 |

**Table xxx – TNS\_MAX\_BANDS in case of 401 and 400 samples per frame**

|  |  |
| --- | --- |
| Sampling rate | TNS\_MAX\_BANDS |
| 48000 | 30 |
| 44100 | 30 |

**Table xxx – TNS\_MAX\_BANDS in case of 1024 and 960 samples per frame**

|  |  |
| --- | --- |
| Sampling rate | TNS\_MAX\_BANDS |
| 48000 | 40 |
| 44100 | 40 |

* + - 1. **Definitions**

|  |  |
| --- | --- |
| **IIF Session** | A IIF Session starts with setting up an IIF Stream with a configuration and lasts until the session is stopped or needs to be reconfigured. |
| **IIF Frame** | A IIF Frame is a portion of the signal belonging to a specific period of time. An IIF Frame has a start time, and end time, and a duration (the difference of end time and start time). The start time of the coded IIF Frame shall be aligned with the first sample of the decoded PCM signal resulting from decoding that IIF Frame. The end time of the coded IIF Frame shall be aligned with the last sample of the decoded PCM signal resulting from decoding that IIF Frame. An IIF Frame is a concept which is not reflected in the syntax. |
| **IIF Block** | A IIF Block is the coded representation of one complete IIF Frame or a part of an IIF Frame and the basic building block of the IIF bitstream. If the coded representation of the audio signals in one IIF Frame is split over multiple IIF Blocks, then those IIF Blocks shall have different values of block\_id. IIF Blocks of subsequent IIF Frames with the same value of block\_id shall carry the same signal (e.g. coded using different bitrates). The decoder can traverse the IIF bitstream block-wise and therefore only process relevant (e.g., for a device) blocks. The block header contains all relevant information about the block, e.g., the block ID and size. |
| **IIF Block Stream** | A IIF Block Stream is comprised of all blocks of the same block ID across all IIF access units. |
| **IIF Access Unit** | An IIF Access Unit is a piece of binary data consisting of one or more IIF Blocks carrying the coded representation of the signal belonging to the current or a previous IIF Frame. One IIF Access Unit contains the full signal presentation of the current IIF Frame, it may contain redundant blocks belonging to a previous IIF Frame.  The presentation time of an IIF Access Unit shall be the presentation time of the current IIF Frame in that IIF Access Unit, which is identified by the IIF Blocks with a block\_priority value of ‘0’ (highest priority). |
| **IIF Packet** | A IIF Packet is a concept defining the unit of transmission over a packet-based network. In the context of this codec, a packet contains either one or multiple complete IIF Access Units, or one or multiple IIF Blocks all belonging to the same IIF Access Unit. |
| **IIF Audio Bus** | The IIF Audio Bus is a concept that allows mapping of incoming audio signals to either a predefined audio presentation (i.e., channels, objects or HOA) and/or to be used by devices in selected roles (e.g. as a driver feed or echo reference).  The signals of all block streams are routed to the IIF Audio Bus ordered by block ID and respective channel order (if blocks carry more than one channel). The IIF Audio Bus supports two sections:   1. Predefined audio presentation represents a full presentation, i.e., channels or objects or HOA. 2. Device specific routing assigns signals to devices and drivers via a routing matrix and assigns roles (i.e., main or auxiliary).   A IIF Session may be configured to only contain a predefined audio presentation, device specific routing or a combination of both.  The flexibility in mapping incoming signals onto the IIF Audio Bus allows for continuous audio signals even if the active channel configuration changes. Therefore, the assignment of IIF Blocks and channel elements to the IIF Audio Bus shall be stable over the duration of a IIF Session, while some audio elements (e.g., channels, objects) or IIF Blocks may drop in or out intentionally, because of channel mode changes, or unintentionally because of packet losses. |

* + - 1. **iif\_specific\_config**

**Table 4.XXY — Syntax of iif\_specific\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_specific\_config(base\_sampling\_rate) |  |  |
| { |  |  |
| iif\_frame\_properties\_config(base\_sampling\_rate); |  |  |
| iif\_audio\_bus\_source\_config(base\_sampling\_rate); |  |  |
| iif\_predef\_audio\_presentation\_config(); |  |  |
| iif\_block\_properties\_config(); |  |  |
| iif\_device\_specific\_routing\_config(); |  |  |
|  |  |  |
| /\* LATENCY INFO \*/ |  |  |
| **max\_interleaving\_depth;** | **3** | **uimsbf** |
|  |  |  |
| **n\_config\_updates;** | **3** | **uimsbf** |
| for (u=0; u<n\_config\_updates; u++) { |  |  |
| **predef\_config\_upd\_idx;** | **3** | **uimsbf** |
| iif\_config\_update (predef\_config\_upd\_idx); |  |  |
| } |  |  |
|  |  |  |
| /\* Additonal features: Tagging and Zoning \*/ |  |  |
| iif\_block\_stream\_tags(); |  |  |
| iif\_block\_stream\_zones(); |  |  |
| } |  |  |

**Table 4.XXY — Syntax of iif\_frame\_properties\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_frame\_properties\_config(base\_sampling\_rate) |  |  |
| { |  |  |
| **frame\_length\_idx;** | **5** | **uimsbf** |
| **b\_window\_switching** | **1** | **uimsbf** |
| if(b\_window\_switching == 0) { |  |  |
| **iif\_window\_shape;** | **2** | **uimsbf** |
| } |  |  |
| **out\_sampling\_rate\_multiplier\_exponent;** | **2** | **uimsbf** |
| out\_sampling\_rate\_multiplier = (1 <<  out\_sampling\_rate\_multiplier\_exponent); |  |  |
| out\_sampling\_rate = base\_sampling\_rate \* out\_sampling\_rate\_multiplier; |  |  |
| } |  |  |

**Table 4.XXY — Syntax of iif\_audio\_bus\_source\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_audio\_bus\_source\_config(base\_sampling\_rate) |  |  |
| { |  |  |
| /\* AUDIO BUS SOURCE INFO \*/ |  |  |
| **audio\_bus\_width\_minus1;** | **8** | **uimsbf** |
| audio\_bus\_width = audio\_bus\_width\_minus1 + 1; |  |  |
| idx = 0; |  |  |
| offs = 0; |  |  |
| audio\_bus\_input\_block[0] = 0; |  |  |
| b\_add\_metadata\_block = 0; |  |  |
| while (offs < audio\_bus\_width || b\_add\_metadata\_block != 0) |  |  |
| { |  |  |
| /\* the number of signals per block is limited by the capacity of the |  |  |
| audio\_payload() \*/ |  |  |
| if(b\_add\_metadata\_block == 0) { |  |  |
| **audio\_signals\_in\_source\_block[idx];** | **5** | **uimsbf** |
| if (audio\_signals\_in\_source\_block[idx] > 0) { |  |  |
| **block\_sampling\_rate\_multiplier\_exponent;** | **2** | **uimsbf** |
| block\_sampling\_rate\_multiplier = (1 <<  block\_sampling\_rate\_multiplier\_exponent); |  |  |
| block\_sampling\_rate[idx] = base\_sampling\_rate \*  block\_sampling\_rate\_multiplier; |  |  |
| for (i=0; i<audio\_signals\_in\_source\_block[idx]; i++) |  |  |
| { |  |  |
| audio\_bus\_input\_block[i+offs] = idx; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
|  |  |  |
| md\_registry\_entries = 0; |  |  |
| **num\_metadata\_configs[idx];** | **4** | **uimsbf** |
| for (m=0; m<num\_metadata\_configs[idx]; m++){ |  |  |
| iif\_metadata\_config(idx, m); |  |  |
| } |  |  |
| **use\_explicit\_metadata\_signaling[idx];** | **1** | **uimsbf** |
|  |  |  |
|  |  |  |
| audio\_bus\_offset\_for\_block[idx] = offs; |  |  |
| offs += audio\_signals\_in\_source\_block[idx]; |  |  |
| idx += 1; |  |  |
| if (offs == audio\_bus\_width) { |  |  |
| **b\_add\_metadata\_block**; | **1** | **uimsbf** |
| } |  |  |
| } |  |  |
| num\_block\_ids = idx; |  |  |
| } |  |  |

**Table 4.XXY — Syntax of iif\_metadata\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_metadata\_config(idx, m) |  |  |
| { |  |  |
| **metadata\_type;** | **8** | **uimsbf** |
| **b\_metadata\_instance\_tag\_present** | **1** | **uimsbf** |
| if (b\_metadata\_instance\_tag\_present) { |  |  |
| **md\_instance\_tag\_minus1;** | **4** | **uimsbf** |
| md\_instance\_tag = md\_instance\_tag\_minus1 + 1; |  |  |
| } else { |  |  |
| md\_instance\_tag = 0; |  |  |
| } |  |  |
|  |  |  |
| md\_entry[idx][m] = find\_md\_settings(); |  |  |
| if (md\_entry[idx][m] == -1) { |  |  |
| md\_entry[idx][m] = md\_registry\_entries; |  |  |
| md\_registry\_entries += 1; |  |  |
|  |  |  |
| metadata\_type[md\_entry[idx][m]] = metadata\_type; |  |  |
| metadata\_instance\_tag[md\_entry[idx][m]] = md\_instance\_tag; |  |  |
|  |  |  |
| **b\_metadata\_is\_byte\_aligned[md\_entry[idx][m]];** | **1** | **uimsbf** |
|  |  |  |
| **b\_metadata\_framing[md\_entry[idx][m]];** | **1** | **uimsbf** |
| if (b\_metadata\_framing[md\_entry[idx][m]]) { |  |  |
| **max\_num\_metadata\_chunks\_code[md\_entry[idx][m]];** | **2** | **uimsbf** |
| **max\_metadata\_spreading\_code[md\_entry[idx][m]];** | **2** | **uimsbf** |
| **b\_md\_synchronized\_to\_video[md\_entry[idx][m]];** | **1** | **uimsbf** |
|  |  |  |
| **md\_frame\_rate\_code[md\_entry[idx][m]];** | **4** | **uimsbf** |
| if (md\_frame\_rate\_code[md\_entry[idx][m] == 15) { |  |  |
| **md\_frame\_rate\_numerator\_minus1** | **32** | **uimsbf** |
| **md\_frame\_rate\_denominator\_minus1;** | **32** | **uimsbf** |
| } |  |  |
| } |  |  |
|  |  |  |
| **b\_metadata\_expiration[md\_entry[idx][m]];** | **1** | **uimsbf** |
| if (b\_metadata\_expiration[md\_entry[idx][m]]) { |  |  |
| **metadata\_expiration\_period\_code[md\_entry[idx][m]];** | **4** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |

**Note:** find\_md\_settings() returns the unique metadata entry index for the current metadata type and instance tag if exists, else -1.

**Table 4.XXY — Syntax of iif\_predef\_audio\_presentation\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_predef\_audio\_presentation\_config() |  |  |
| { |  |  |
| CHANNEL\_MASK\_BITS = [15,32,43,128]; |  |  |
|  |  |  |
| num\_signals\_for\_audio\_presentation = 0; |  |  |
| channel\_mask\_superset = 0; |  |  |
| num\_hoa\_signals = 0; |  |  |
| object\_metadata\_id = 0; |  |  |
| num\_audio\_objects = 0; |  |  |
|  |  |  |
| **audio\_presentation\_type;** | **2** | **uimsbf** |
| switch (audio\_presentation\_type) { |  |  |
| case AUDIO\_PRESENTATION\_TYPE\_CHANNELS: |  |  |
| **channel\_mode\_superset;**  /\* according to ISO/IEC 23091-3:2018, Table 3 \*/ | **5** | **uimsbf** |
| if (channel\_mode\_superset == 0) { |  |  |
| **channel\_mask\_width;** | **2** | **uimsbf** |
| **channel\_mask\_superset;**  /\* according to ISO/IEC 23091-3:2018, Table-2 \*/ | **CHANNEL\_MASK\_BITS[channel\_mask\_width]** | **uimsbf** |
| } |  |  |
| else { |  |  |
| channel\_mask\_superset =  get\_channel\_mask\_for\_channel\_config  (channel\_mode\_superset); |  |  |
| } |  |  |
| num\_audio\_channels =  count\_audio\_channels(channel\_mask\_superset); |  |  |
| num\_signals\_for\_audio\_presentation += num\_audio\_channels; |  |  |
|  |  |  |
| channel\_config\_set\_channel\_mode[0] = channel\_mode\_superset; |  |  |
| channel\_config\_set\_channel\_mask[0] = channel\_mask\_superset; |  |  |
| channel\_config\_set\_mix\_matrix[0] =  init\_mix\_matrix(channel\_mask\_superset); |  |  |
|  |  |  |
| **num\_channel\_config\_subsets;** | **3** | **uimsbf** |
| for (i=0; i<num\_channel\_config\_subsets; i++) { |  |  |
| **channel\_config\_set\_channel\_mode[i+1];**  /\* according to ISO/IEC 23091-3:2018, Table 3 \*/ | **5** | **uimsbf** |
| if (channel\_config\_set\_channel\_mode[i+1] == 0) { |  |  |
| **channel\_mask\_width;** | **2** | **uimsbf** |
| **channel\_config\_set\_channel\_mask[i+1];**  /\* according to ISO/IEC 23091-3:2018, Table-2 \*/ | **CHANNEL\_MASK\_BITS[channel\_mask\_width]** | **uimsbf** |
| } |  |  |
| else { |  |  |
| channel\_config\_set\_channel\_mask[i+1] = |  |  |
| get\_channel\_mask\_for\_channel\_config  (channel\_config\_set\_channel\_mode[i+1]); |  |  |
| } |  |  |
|  |  |  |
| channel\_config\_set\_mix\_matrix[i+1] =  init\_mix\_matrix(channel\_mask\_superset); |  |  |
| channels\_not\_in\_super =  channel\_config\_set\_channel\_mask[i+1] -  (channel\_config\_set\_channel\_mask[i+1] &  channel\_mask\_superset); |  |  |
| if (channels\_not\_in\_super > 0) { |  |  |
| channel\_subset\_mix\_info(channels\_not\_in\_super, i+1); |  |  |
| } |  |  |
| } |  |  |
| break; |  |  |
|  |  |  |
| case AUDIO\_PRESENTATION\_TYPE\_HOA: |  |  |
| **hoa\_order\_minus\_one;** | **3** | **uimsbf** |
| num\_hoa\_signals = 1 << (hoa\_order\_minus\_one + 2); |  |  |
| num\_signals\_for\_audio\_presentation += num\_hoa\_signals; |  |  |
| break; |  |  |
|  |  |  |
| case AUDIO\_PRESENTATION\_TYPE\_OBJECTS: |  |  |
| **object\_metadata\_id;** | **8** | **uimsbf** |
| **num\_audio\_objects\_minus1;** | **8** | **uimsbf** |
| num\_audio\_objects = num\_audio\_objects\_minus\_one + 1; |  |  |
| num\_signals\_for\_audio\_presentation += num\_audio\_objects; |  |  |
| break; |  |  |
|  |  |  |
| case AUDIO\_PRESENTATION\_TYPE\_NONE: |  |  |
| break; |  |  |
| } |  |  |
| } |  |  |

**Table 4.XXY — Syntax of iif\_block\_properties\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_block\_properties\_config() |  |  |
| { |  |  |
| num\_signals\_for\_device\_routing = audio\_bus\_width –  num\_signals\_for\_audio\_presentation; |  |  |
| if (num\_signals\_for\_device\_routing > 0) { |  |  |
| **num\_target\_devices;** | **4** | **uimsbf** |
|  |  |  |
| if (num\_target\_devices > 0) { |  |  |
| **b\_enable\_device\_addressing;** | **1** | **uimsbf** |
| /\* num\_blocks\_ids is already defined at Audio Bus Source Info \*/ |  |  |
| for (i=0; i<num\_block\_ids; i++) { |  |  |
| if (b\_enable\_device\_addressing == 1) { |  |  |
| **b\_address\_to\_all\_devices[i];**  /\* add a target adressing mask, 0=all targets \*/ | **1** | **uimsbf** |
| if (b\_address\_to\_all\_devices[i] == 0) { |  |  |
| **device\_address\_mask[i];** | **num\_target\_devices bits** |  |
| } |  |  |
| else { |  |  |
| device\_address\_mask[i] = (1<<num\_target\_devices)  – 1; |  |  |
| } |  |  |
| } |  |  |
| else { |  |  |
| b\_address\_to\_all\_devices[i] = 1; |  |  |
| device\_address\_mask[i] = (1<<num\_target\_devices) - 1; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| else { |  |  |
| b\_enable\_device\_addressing = 0; |  |  |
| } |  |  |
| } |  |  |
| else { |  |  |
| num\_target\_devices = 0; /\* this implies broadcast mode \*/ |  |  |
| } |  |  |
|  |  |  |
| } |  |  |

**Table 4.XXY — Syntax of iif\_device\_specific\_routing\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_device\_specific\_routing\_config() |  |  |
| { |  |  |
| for (d=0; d<num\_target\_devices; d++) { |  |  |
| for (s=0; s<num\_signals\_for\_device\_routing; s++) { |  |  |
| \_offs = num\_signals\_for\_audio\_presentation + s; |  |  |
| \_block = audio\_bus\_input\_block[\_offs]; |  |  |
| audio\_signal\_bus\_offs[d][s] = \_offs; |  |  |
| if (device\_address\_mask[\_block] & (1<<d)) { |  |  |
| /\* data are only present for audio signals on the bus |  |  |
| delivered by blocks addressed to device d \*/ |  |  |
| bus\_to\_device\_mapping\_defaults(d,s); |  |  |
| } |  |  |
| else { |  |  |
| default\_audio\_signal\_role[d][s] =  AUDIO\_SIGNAL\_NO\_ROLE; |  |  |
| default\_audio\_signal\_gain[d][s] = 0; |  |  |
| default\_audio\_signal\_delay[d][s] = 0; |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 4.XXY — Syntax of bus\_to\_device\_mapping\_defaults**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| bus\_to\_device\_mapping\_defaults(device, bus) |  |  |
| { |  |  |
| **b\_default\_signal\_connected;** | **1** | **uimsbf** |
| if (b\_default\_signal\_connected == 1) { |  |  |
| **default\_audio\_signal\_role[device][bus];** | **3** | **uimsbf** |
| } |  |  |
| else { |  |  |
| default\_audio\_signal\_role[device][bus] = AUDIO\_SIGNAL\_NO\_ROLE; |  |  |
| } |  |  |
| /\* allow setting defaults for gain and delay for all signals, |  |  |
| even if not connected by default \*/ |  |  |
| default\_audio\_signal\_gain[device][bus] = 0; |  |  |
| **b\_set\_default\_custom\_gain;** | **1** | **uimsbf** |
| if (b\_set\_default\_custom\_gain) { |  |  |
| **gain\_code;** | **5** | **uimsbf** |
| if (gain\_code < 24) { |  |  |
| default\_audio\_signal\_gain[device][bus] = gain\_code - 24; |  |  |
| } |  |  |
| else { |  |  |
| default\_audio\_signal\_gain[device][bus] = gain\_code - 23; |  |  |
| } |  |  |
| } |  |  |
| default\_audio\_signal\_delay[device][bus] = 0; |  |  |
| **b\_set\_default\_custom\_delay;** | **1** | **uimsbf** |
| if (b\_set\_default\_custom\_delay) { |  |  |
| **delay\_code;** | **12** | **uimsbf** |
| default\_audio\_signal\_delay[device][bus] = 4\*delay\_code; |  |  |
| } |  |  |
| } |  |  |

**Table 4.XXY — Syntax of iif\_block\_stream\_tags**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_block\_stream\_tags() |  |  |
| { |  |  |
| for (blk = 0; blk < num\_block\_ids; ++blk) { |  |  |
| **b\_block\_tags\_present;** | **1** | **uimsbf** |
| if (b\_block\_tags\_present) { |  |  |
| **num\_block\_tags\_minus1** | **8** | **uimsbf** |
| for (i = 0; i < num\_block\_tags\_minus1 + 1; ++i) { |  |  |
| **block\_tag\_id;** | **8** | **uimsbf** |
| **block\_tag\_size;** | **4** | **uimsbf** |
| if (block\_tag\_size > 0) { |  |  |
| **block\_tag\_value;** | **8\*block\_tag\_size** | **uimsbf** |
| } |  |  |
| **b\_tag\_allows\_control;** | **1** | **uimsbf** |
| if (block\_tag\_size > 0 && b\_tag\_allows\_control) { |  |  |
| **b\_mutually\_exclusive\_playout;** | **1** | **uimsbf** |
| } |  |  |
| } |  |  |
| **b\_default\_skip;** | **1** | **uimsbf** |
| } |  |  |
| }  } |  |  |

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_block\_stream\_zones() |  |  |
| { |  |  |
| for (blk = 0; blk < num\_block\_ids; ++blk) { |  |  |
| has\_zone\_exclusion\_signalling = 0; |  |  |
| for (t = 0; t <= num\_block\_tags\_minus1[blk]; ++t) { |  |  |
| if (block\_tag\_id[blk][t] == 14 || // All zones  block\_tag\_id[blk][t] == 16) { // User |  |  |
| **has\_zone\_exclusion\_signalling;** | **1** | **uimsbf** |
| break;  }  } |  |  |
| if ( has\_zone\_exclusion\_signaling) { |  |  |
| **num\_excluded\_zones;** | **4** | **uimsbf** |
| for(j=0;j<num\_excluded\_zones;j++) { |  |  |
| **block\_tag\_value[blk][j];** | **8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

**Table 4.XXY — Syntax of channel\_subset\_mix\_info**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| channel\_subset\_mix\_info(\_channels\_not\_in\_super\_arg, \_set) |  |  |
| { |  |  |
| src\_ch\_idx = 0; |  |  |
| \_channels\_not\_in\_super = \_channels\_not\_in\_super\_arg; |  |  |
| while (\_channels\_not\_in\_super > 0) { |  |  |
| if ((\_channels\_not\_in\_super & 1) == 0x1) { |  |  |
| **map\_or\_mix;** | **1** | **uimsbf** |
| **is\_channel\_pair;** | **1** | **uimsbf** |
| n\_mixing\_coeffs = map\_or\_mix + 1; |  |  |
| for (m=0; m<n\_mixing\_coeffs; m++) { |  |  |
| **upmix\_target\_ch\_idx;** | **7** | **uimsbf** |
| **upmix\_gain;** | **3** | **uimsbf** |
|  |  |  |
| ch\_offs = 0; |  |  |
| \_ch\_mask = channel\_mask\_superset; |  |  |
| \_ch\_idx = upmix\_target\_ch\_idx; |  |  |
| while (\_ch\_idx > 0) |  |  |
| { |  |  |
| if ((\_ch\_mask & 0x1) == 0x1) { |  |  |
| ch\_offs += 1; |  |  |
| } |  |  |
| \_ch\_mask = \_ch\_mask >> 1; |  |  |
| \_ch\_idx = \_ch\_idx - 1; |  |  |
| } |  |  |
|  |  |  |
|  |  |  |
| channel\_config\_set\_mix\_matrix[\_set][ch\_offs][3\*m+0] =  src\_ch\_idx; |  |  |
| channel\_config\_set\_mix\_matrix[\_set][ch\_offs][3\*m+1] =  upmix\_target\_ch\_idx; |  |  |
| channel\_config\_set\_mix\_matrix[\_set][ch\_offs][3\*m+2] =  upmix\_gain; |  |  |
| if (is\_channel\_pair == 1) { |  |  |
| ch\_offs += 1; |  |  |
| channel\_config\_set\_mix\_matrix[\_set][ch\_offs][3\*m+0] =  src\_ch\_idx+1; |  |  |
| channel\_config\_set\_mix\_matrix[\_set][ch\_offs][3\*m+1] =  upmix\_target\_ch\_idx+1; |  |  |
| channel\_config\_set\_mix\_matrix[\_set][ch\_offs][3\*m+2] =  upmix\_gain; |  |  |
| } |  |  |
| } |  |  |
| if (is\_channel\_pair == 1) { |  |  |
| \_channels\_not\_in\_super = \_channels\_not\_in\_super >> 1; |  |  |
| src\_ch\_idx += 1; |  |  |
| } |  |  |
| } |  |  |
| \_channels\_not\_in\_super = \_channels\_not\_in\_super >> 1; |  |  |
| src\_ch\_idx += 1; |  |  |
| } |  |  |
| } |  |  |

**Table 4.XXY — Syntax of init\_mix\_matrix**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| init\_mix\_matrix(\_channel\_mask\_superset\_arg) |  |  |
| { |  |  |
| \_ch\_cnt = 0; |  |  |
| \_ch\_idx = 0; |  |  |
| \_mix\_mtrx = []; |  |  |
| \_channel\_mask\_superset = \_channel\_mask\_superset\_arg; |  |  |
| while (\_channel\_mask\_superset > 0) { |  |  |
| if ((\_channel\_mask\_superset & 0x1) == 0x1) { |  |  |
| \_mix\_mtrx[\_ch\_cnt] = [\_ch\_idx, \_ch\_idx, 0]; |  |  |
| \_ch\_cnt += 1; |  |  |
| } |  |  |
| \_channel\_mask\_superset = \_channel\_mask\_superset >> 1; |  |  |
| \_ch\_idx += 1; |  |  |
| } |  |  |
| return \_mix\_mtrx; |  |  |
| } |  |  |

**Table 4.XXY — Syntax of count\_audio\_channels**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| count\_audio\_channels(\_channel\_mask) |  |  |
| { |  |  |
| \_audio\_channels = 0; |  |  |
| \_ch\_mask = \_channel\_mask; |  |  |
| while (\_ch\_mask > 0) { |  |  |
| \_audio\_channels += (\_ch\_mask & 0x1); |  |  |
| \_ch\_mask = \_ch\_mask >> 1; |  |  |
| } |  |  |
| return \_audio\_channels; |  |  |
| } |  |  |

**Table 4.XXY — Syntax of get\_channel\_mask\_for\_channel\_config**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| get\_channel\_mask\_for\_channel\_config(\_channel\_config) |  |  |
| { |  |  |
| channel\_mask\_for\_channel\_mode = [ |  |  |
| 0, /\* 0: \*/ |  |  |
| 0b100, /\* 1: C (1.0) \*/ |  |  |
| 0b11, /\* 2: L,R (2.0) \*/ |  |  |
| 0b111, /\* 3: L,R,C (3.0) \*/ |  |  |
| 0b10000000111, /\* 4: L,R,C,Cs \*/ |  |  |
| 0b110111, /\* 5: L,R,C,Ls,Rs (5.0) \*/ |  |  |
| 0b111111, /\* 6: L,R,C,LFE,Ls,Rs (5.1) \*/ |  |  |
| 0b11111111, /\* 7: L,R,C,LFE,Ls,Rs,Lc,Rc \*/ |  |  |
| 0, /\* 8: Dual-mono is unsupported! \*/ |  |  |
| 0b10000000011, /\* 9: L,R,Cs \*/ |  |  |
| 0b110011, /\* 10: L,R,Ls,Rs \*/ |  |  |
| 0b10000111111, /\* 11: L,R,C,LFE,Ls,Rs,Cs \*/ |  |  |
| 0b1100111111, /\* 12: L,R,C,LFE,Ls,Rs,Lsr,Rsr (7.1) \*/ |  |  |
| 0b00111111111111100110011111001111, /\* 13: L,R,C,LFE,Lc,Rc,Lsr,Rsr,Cs,Lss,Rss,Lv,Rv,Cv,Lvr,Rvr,Cvr,Lvss,Rvss,Ts,LFE2,Lb,Rb,Cb (22.2) \*/ |  |  |
| 0b1100000000000111111, /\* 14: L,R,C,LFE,Ls,Rs,Lv,Rv \*/ |  |  |
| 0b100010001100110000000111111, /\* 15: L,R,C,LFE,Ls,Rs,Lss,Rss,Lv,Rv,Cvr,LFE2 \*/ |  |  |
| 0b11000000000001100000000000111111, /\* 16: L,R,C,LFE,Ls,Rs,Lv,Rv,Lvs,Rvs (5.1.4) \*/ |  |  |
| 0b11000010000011100000000000111111, /\* 17: L,R,C,LFE,Ls,Rs,Lv,Rv,Cv,Ts,Lvs,Rvs \*/ |  |  |
| 0b1100000000011000010000011100000000000111111, /\* 18: L,R,C,LFE,Ls,Rs,Lv,Rv,Cv,Ts,Lvs,Rvs,Lbs,Rbs \*/ |  |  |
| 0b1101100110001100001111, /\* 19: L,R,C,LFE,Lsr,Rsr,Lss,Rss,Lv,Rv,Lvr,Rvr (7.1.4) \*/ |  |  |
| 0b110000011000000000001100110001100001111, /\* 20: L,R,C,LFE,Lsr,Rsr,Lss,Rss,Lv,Rv,Lvs,Rvs,Leos,Reos \*/ |  |  |
| 0, /\* 21: Reserved \*/ |  |  |
| 0, /\* 22: Reserved \*/ |  |  |
| 0, /\* 23: Reserved \*/ |  |  |
| 0, /\* 24: Reserved \*/ |  |  |
| 0, /\* 25: Reserved \*/ |  |  |
| 0, /\* 26: Reserved \*/ |  |  |
| 0, /\* 27: Reserved \*/ |  |  |
| 0, /\* 28: Reserved \*/ |  |  |
| 0, /\* 29: Reserved \*/ |  |  |
| 0, /\* 30: Reserved \*/ |  |  |
| 0 /\* 31: Reserved \*/ |  |  |
| ]; |  |  |
|  |  |  |
| \_ch\_mask2 = 0; |  |  |
| if (\_channel\_config < 32) { |  |  |
| \_ch\_mask2 = channel\_mask\_for\_channel\_mode[\_channel\_config]; |  |  |
| } |  |  |
| return \_ch\_mask2; |  |  |
| } |  |  |

* + - 1. **Semantics of iif\_specific\_config**

|  |  |
| --- | --- |
| **frame\_length\_idx** | Indicates the frame length in audio samples for the IIF stream, according to Table 4.xxx. |
| **Table 4.xxx – Values of the frame\_length\_idx field**   |  |  | | --- | --- | | Value of frame\_length\_idx | IIF frame length in samples | | 0 | 120 | | 1 | 240 | | 2 | 480 | | 3 | 960 | | 4 | 128 | | 5 | 256 | | 6 | 512 | | 7 | 1024 | | 8 | 192 | | 9 | 384 | | 10 | 768 | | 11-15 | Reserved for ISO use | | 16 | 200 | | 17 | 200.2 | | 18 | 250 | | 19 | 250.25 | | 20 | 400 | | 21 | 400.4 | | 22 | 500 | | 23 | 500.5 | | 24-31 | Reserved for ISO use | | |
|  | |
| **b\_window\_switching** | Indicates whether window shape switching is enabled. If disabled (0), window\_shape transmitted in ics\_info() shall be ignored and only iif\_window\_shape shall be used. |
| **iif\_window\_shape** | |  |  | | --- | --- | | Value of iif\_window\_shape | Window shape | | 0 | sine window as defined in 4.6.21.4 (window\_shape equal to ‘0’) | | 1 | low-overlap window as defined in 4.6.21.4 (window\_shape equal to ‘1’) | | 2 | asymmetric window; shall only be used for frame\_length\_idx equal to ‘1’ or ‘2’ (see below) | | 3 | Reserved for ISO use |   If iif\_window\_shape is set to ‘2’ (asymmetric window), the coefficients stored in the following tables shall be used, depending on the value of frame\_length\_idx:   |  |  | | --- | --- | | Value of frame\_length\_idx | Table | | 1 | 4.A.92 | | 2 | 4.A.93 |   *Note: The values given are for the decoding process, for encoding the window is time-reversed.* |
| **out\_sampling\_rate\_multiplier\_exponent** | Indicates the exponent to the base of 2 of the out\_sampling\_rate\_multiplier to be applied to the base\_sampling\_rate. For values of ‘0’-‘2’, the output sampling rate shall be calculated as follows:  The value of 3 shall not be used. |
| **audio\_bus\_width\_minus1** | Plus 1 indicates the width of the IIF Audio Bus. |
| **audio\_signals\_in\_source\_block** | Indicates the number of audio signals delivered by the IIF block with the block\_id matching the counting index. A value of ‘0’ (zero) indicates that a block does not carry any audio data. |
| **block\_sampling\_rate\_multiplier\_exponent** | Indicates the exponent to the base of 2 of the block\_sampling\_rate\_multiplier to be applied to the base\_sampling\_rate. This value shall not exceed the value of the out\_sampling\_rate\_multiplier\_exponent element.  For values of ‘0’-‘2’, the output sampling rate shall be calculated as follows:  The value of 3 shall not be used. |
| **num\_metadata\_config** | Indicates the number of metadata configurations. |
| **metadata\_type** | Indicates the type of the metadata as according to Table 4.xxx. |
| **Table 4.xxx – Values of the metadata\_type field**   |  |  | | --- | --- | | Value of metadata\_type | Metadata type | | 0 | METADATA\_TYPE\_AUTH | | 1-127 | Reserved for ISO use. | | 128-255 | Reserved for use outside of ISO scope | | |
|  |  |
| **b\_metadata\_instance\_tag\_present** | Indicates the presence of a metadata instance tag. |
| **md\_instance\_tag\_minus1** | The metadata instance tag minus 1. Metadata with the same type and instance tag belong together. They need to be concatenated across blocks within a frame according to ascending order of the block\_id and across frames in strict time order of the frames. |
| **b\_metadata\_is\_byte\_aligned** | Indicates if the metadata are byte-aligned. |
| **b\_metadata\_framing** | Indicates that metadata use the framing provided by the codec and are not implicitly aligned to the same framing as the audio. |
| **max\_num\_metadata\_chunks\_code** | Indicates the maximum number of metadata chunks per section for this type of metadata. It also specifies the number of bits used to signal the number of chunks in a block.   |  |  |  | | --- | --- | --- | | Value of max\_num\_metadata\_chunks\_code | max\_chunk\_bits | Maximum number of metadata chunks | | 0 | 1 | 2 | | 1 | 2 | 4 | | 2 | 3 | 8 | | 3 | 4 | 16 | |
| **max\_metadata\_spreading\_code** | Indicates the maximum number of IIF metadata chunks a single metadata frame can be spread across. It also specifies the number of bits used to signal the chunk.   |  |  |  | | --- | --- | --- | | Value of max\_num\_metadata\_spreading\_code | max\_spreading\_bits | Maximum number of chunks | | 0 | 4 | 16 | | 1 | 5 | 32 | | 2 | 6 | 64 | | 3 | 8 | 256 | |
| **b\_md\_synchronized\_to\_video** | If set, this bit indicates that the metadata frame rate matches the accompanying video frame rate, and the metadata frames are synchronized to video frames. This means the first sample for which the metadata is valid is aligned with a video frame boundary. |
| **md\_frame\_rate\_code** | Indicates the frame rate of the metadata according to Table 4.xxx. |
| **Table 4.xxx – Values of the md\_frame\_rate\_code field**   |  |  | | --- | --- | | Value of md\_frame\_rate\_code | Frame rate in frames per second | | 0 | Undefined | | 1 | 23.98 | | 2 | 24 | | 3 | 25 | | 4 | 29.97 drop-frame (DF) | | 5 | 29.97 non-drop-frame (NDF) | | 6 | 30 | | 7 | 50 | | 8 | 59.94 DF | | 9 | 59.94 NDF | | 10 | 60 | | 11 | 100 | | 12 | 119.88 DF | | 13 | 119.88 NDF | | 14 | 120 | | 15 | Escape value | | |
| **md\_frame\_rate\_numerator\_minus1** | See **md\_frame\_rate\_denominator\_minus1**. |
| **md\_frame\_rate\_denominator\_minus1** | The metadata frame rate shall be calculated as: |
| **b\_metadata\_expiration** | Signal that metadata can be kept for a longer period if updates are missing. If unset (0), metadata is only applicable to current frame. |
| **metadata\_expiration\_period\_code** | Signal a metadata expiration period. This period starts at the end of the frame containing that last metadata chunk.   * 0-14: (range: 20ms – 63.246s) * 15: no expiration |
| **use\_explicit\_metadata\_signaling** | Indicates if metadata are signaled explicitly. |
| **b\_add\_metadata\_block** | Indicates that another block stream is configured for metadata only. |
| **num\_block\_ids** | Indicates the number of block streams. |
| **num\_excluded\_zones** | Number of excluded zones associated with the block stream. |
| **audio\_presentation\_type** | Indicates the type of the audio presentation according to the Table 4.xxx. |
| **Table 4.xxx – Values of the audio\_presentation\_type field**   |  |  |  | | --- | --- | --- | | Symbol | Value of audio\_presentation\_type | Purpose | | AUDIO\_PRESENTATION\_TYPE\_NONE | ‘00’ | No signals are assigned to a predefined audio presentation. | | AUDIO\_PRESENTATION\_TYPE\_CHANNELS | ‘01’ | Audio presentation type channels. | | AUDIO\_PRESENTATION\_TYPE\_HOA | ’10‘ | Audio presentation type HOA. | | AUDIO\_PRESENTATION\_TYPE\_OBJECTS | ‘11’ | Audio presentation type objects. | | |
|  |  |
| **channel\_mode\_superset** | Indicates the channel mode superset according to ISO/IEC 23091-3:2018, Table 3. |
| **channel\_mask\_width** | Identifies the number of least significant bits signaled for the channel mask according to Table 4.xxx. All bits with higher significance shall be ‘0’. |
| **Table 4.xxx – Values of the channel\_mask\_width field**   |  |  | | --- | --- | | Value of channel\_mask\_width | Number of least significant bits | | 0 | 15 | | 1 | 32 | | 2 | 43 | | 3 | 128 | | |
|  |  |
| **channel\_mask\_superset** | Indicates selected channels via a bitmask. Selected channels are indicated by setting the respective bit of the bitmask to a value of ‘1’ in accordance with ISO/IEC 23091-3:2018, Table-2, where the significance of the bit represents the “Value” of the respective channel in ISO/IEC 23091-3:2018, Table-2. |
| **num\_channel\_config\_subsets** | Indicates the number of channel configuration subsets. |
| **channel\_config\_set\_channel\_mode** | Indicates the channel mode for the respective channel config set, according to ISO/IEC 23091-3:2018, Table 3. |
| **channel\_config\_set\_channel\_mask** | Indicates selected channels via a bitmask. Selected channels are indicated by setting the respective bit of the bitmask to a value of ‘1’ in accordance with ISO/IEC 23091-3:2018, Table-2, where the significance of the bit represents the “Value” of the respective channel in ISO/IEC 23091-3:2018, Table-2. |
| **hoa\_order\_minus1** | Plus 1 indicates the order of the higher order ambisonics (HOA) signal. |
| **object\_metadata\_id** | Indicates an identifier according to Table 4.xxx that can be used to map metadata (e.g., from the iif\_metadata\_frame syntax element) to the related object. |
| **num\_audio\_objects\_minus\_one** | Plus 1 indicates the number of audio objects. |
| **num\_target\_devices** | Indicates the number of target devices. |
| **b\_enable\_device\_addressing** | Indicates if device addressing is enabled. |
| **b\_address\_to\_all\_devices** | Indicates if the block stream with the block\_id=i is addressed to all devices. |
| **device\_address\_mask** | Indicates the device address mask for the respective block\_id. Bit 0 of this mask refers to the least significant bit of that mask and is associated with the device\_id=’0’, bit 1 with device\_id=’1’, and other bits respectively. |
| **max\_interleaving\_depth** | Indicates the maximum interleaving depth of redundant blocks in the stream of blocks. |
| **n\_config\_updates** | Specifies the total number of predefined config updates. |
| **predef\_config\_upd\_idx** | Specifies the index of a particular predefined config update. A value of ‘0’ is reserved for inband configuration updates. |
| **b\_default\_signal\_connected** | Indicates if a signal is connected and a role is assigned on a specific device.  A value of ‘0’ indicates that the audio signal on default\_audio\_signal\_bus\_offs[d][s] is not connected on the device d. Its role shall be set to AUDIO\_SIGNAL\_NO\_ROLE.  A value of ‘1’ indicates that the audio signal on default\_audio\_signal\_bus\_offs[d][s] is connected on the device d. Its role shall be transmitted explicitly. |
| **default\_audio\_signal\_role** | Indicates the default role of the audio signal according to Table 4.ASR. |
| **b\_set\_default\_custom\_gain** | Indicates if a default custom gain is signaled. |
| **gain\_code** | See 4.5.2.18.1. |
| **b\_set\_default\_custom\_delay** | Indicates if a default custom delay is signaled. |
| **delay\_code** | See 4.5.2.18.1. |
| **b\_block\_tags\_present** | Indicates whether a block stream has tags assigned or not. If equal to 1, a conforming decoder shall provide input and output interfaces as defined in Table 4.XXY. |
| **num\_block\_tags\_minus1** | Specifies the number of tags assigned to a block stream. The number of assigned tags = num\_block\_tags\_minus1 + 1. |
| **block\_tag\_id** | The block tag ID as defined in Table 4.93 for further processing of the IIF block stream. Each block\_tag\_id shall occur at most once in a block except for block\_tag\_id 15, 16. For time critical sounds. (10...12), the following restrictions apply:   * For the given block stream, the max\_interleaving\_depth shall be treated as having value 0. * Any non-mandatory additional processing adding delay (e.g. an optional limiter with lookahead) in the decoder should be disabled for processing this block stream. |
| **Table 4.93 – block\_** **tag\_id mapping**   |  |  |  | | --- | --- | --- | | block\_tag\_id value | block\_tag\_size | Definition | | 0 | as signaled | reserved | | 1 | 0 | Clean/Discrete Audio Description. Defines that the signal is supposed to be played out if an audio description for visually impaired persons is requested. Tagged signal contains clean audio description, not mixed with other sounds. | | 2 | 0 | Audio Description Mix. Defines that the signal is supposed to be played out if an audio description for visually impaired persons is requested.  Tagged signal contains a full mix consisting of audio description and other sounds. | | 3 | 3 | Language code of the signal coded as 3-character code as specified in ISO 639-2. | | 4 | 0 | Pre-Binauralizaed, indicates that the audio signal being transmitted has been already binauralized and thus should not be binauralized again. | | 5 | 0 | Upmixed, indicates that the audio signal being transmitted has been already upmixed and thus should not be upmixed/downmixed again. | | 6 | 0 | Original mix, indicates that the audio signal being transmitted should only be postprocessed to a level required for playback, in other words, any optional postprocessing should be skipped. | | 7 | 0 | Dialog/commentary, indicates that the audio signal being transmitted contains only dialog/commentary. This information may be used for dialog/commentary specific processing. | | 8 | 0 | Natural language interface/voice message | | 9 | 0 | Instructive/informative sound (e.g. doorbell) | | 10 | 0 | System sound, e.g. acoustical feedback when operating the remote control. | | 11 | 0 | Emergency sound priority 1 (mandatory). | | 12 | 0 | Emergency sound priority 2 (optional). | | 13 | as signaled | Text label in UTF-8 encoding according to ISO/IEC 10646 | | 14 | 0 | Zone 0: “user" – block stream shall only be decoded in all zones with one or more users present and where the zone is not excluded (has\_zone\_exclusion\_signalling = 1). | | 15 | 1 | Zone identifier – block stream shall be decoded in signaled zone. | | 16 | 1 | User identifier – block stream shall only be decoded in zones where the signaled user is present and where the zone is not excluded (has\_zone\_exclusion\_signalling = 1). | | 17..255 | as signaled | reserved | | |
| **block\_tag\_size** | Specifies the size of the block\_tag\_value in bytes. |
| **block\_tag\_value** | Indicate the value of a block tag for the transmitted block\_tag\_id as defined in Table 4.93. |
| **b\_tag\_allows\_control** | Indicates if a tag can be used to filter signals (decode or skip), or if it only carries informative data. The value of b\_tag\_allows\_control shall be identical for all tags with same block\_tag\_id in the stream. |
| **b\_mutually\_exclusive\_playout** | If set to 1, out of all block streams tagged with this tag, exclusively the block stream tagged with one block tag value shall be played out at a time. The value of b\_mutually\_exclusive\_playout shall be identical for all tags with same block\_tag\_id in the stream. If b\_mutually\_exclusive\_playout==1 for a block\_tag\_id of a block, then in the stream the value of b\_default\_skip shall be 0 for exactly one block with this block\_tag\_id. |
| **b\_default\_skip** | Defines the default signal filtering (decode or skip) state/behavior. The default can be overridden by the decoder control interface. The value of default\_skip shall be 0 if there is no tag with b\_tag\_allows\_control==1 for the current block. |
| **map\_or\_mix** | Indicates if a source channel indicated by a bit in the respective channel\_config\_set\_channel\_mask that is not present in the channel\_mask\_superset is mapped at another channel in the superset (0) or upmixed into two channels of the superset. |
| **is\_channel\_pair** | Indicates if the following mapping or mixing parameters are applied to a single channel (0) or to two consecutive channels (1). |
| **upmix\_target\_ch\_idx** | Indicates the channel offset in the superset the source channel has been mapped or mixed into before encoding. This corresponds to the channel offset on the audio bus. |
| **upmix\_gain** | Indicates the gain that has been applied to the source signal before mapping or mixing it into the target channel of the superset according to Table 4.MIXGAIN. |
| **Table 4.MIXGAIN – Values of the upmix\_gain field**   |  |  | | --- | --- | | Value of upmix\_gain | Mixing gain | | 0 | 1.0 (0 dB) | | 1 | (~-1.5 dB) | | 2 | (~-3 dB) | | 3 | (~-4.5 dB) | | 4 | (~-6 dB) | | 5 | (~-9 dB) | | 6 | (~-12 dB) | | 7 | reserved | | |
|  |  |

* + - 1. **Decoding process**
         1. **Variables**

|  |  |
| --- | --- |
| block\_bytes\_left | Indicates the number of bytes remaining for the current block, excluding **block\_crc (**if **b\_block\_protected** == 1). When used in the syntax, it is assumed that this value is always up to date. |
| metadata\_section\_type | An array populated with the metadata types transmitted in the current block. Metadata types are defined in Table 4.xxx – Values of the metadata\_type field |
| num\_metadata\_sections | Indicates the number of metadata sections transmitted in the current block (corresponds to the size of the array metadata\_section\_type). |
| n\_metadata\_bits | Indicates the number of bits of one metadata section. |
| active\_channel\_mode | Indicates the currently active channel mode subset. |
| active\_channel\_mask | Indicates the currently active channel mask subset. |
|  |  |

* + - * 1. **Decoding of iif\_specific\_config**

As an extension to the abstract class AudioSpecificConfig(), the iif\_specific\_config() carries IIF specific info enabling signaling for specific use cases, e.g., broadcasting. It provides information such as signals to devices mapping enabling a particular device to skip over irrelevant parts and only decode relevant IIF Blocks.

Static parameters such as the IIF Frame length and the intended output sampling rate are decoded in iif\_frame\_properties\_config(). The signaled frame length and sampling rate shall be used to configure the underlying tools of the low delay codec. The iif\_audio\_bus\_source\_config() provides the mapping of an IIF Block containing audio signals into the IIF Audio Bus and carries the information on the metadata config and signaling of the corresponding IIF Block. The amount of IIF Blocks, num\_block\_ids, being mapped into the IIF Audio Bus is limited by the decoded audio bus width value, i.e., the total amount of signals in an IIF Audio Bus. At this point, the mapping information of each signal in the audio bus at index [i] to the corresponding IIF Block ID [block\_id] is shown as audio\_bus\_input\_block[i] = block\_id. Note that the specific IIF Block sampling rate is also specified here.

For each block\_id, the presence of metadata configs is signaled. If present, information on the metadata type, metadata instance tag, metadata framing/chunking and metadata expiration are signaled in iif\_metadata\_config(block\_id, m) for all values of m, denoting the number of metadata configs for a particular block\_id. Additionally, the use of an explicit metadata signaling for a particular block\_id is also specified. This is to allow an explicit metadata config index assignment during an IIF Block decoding process.

Next, iif\_predef\_audio\_presentation\_config() enables the mapping of a subset of signals in IIF Audio Bus into a predefined audio presentation layout, i.e., channels, objects, or HOA. Following the decoding operation, the total number of signals consumed by this mapping is denoted by num\_signals\_for\_audio\_presentation. If there is a remaining subset of audio signals, it provides an indication that the (subset of) signals are to be broadcast to all devices or specifically routed to a list of N numbers of target devices, num\_target\_devices. The indication is obtained following the decoding operation in iif\_block\_properties\_config(). If a device routing is indicated, a device\_address\_mask[block\_id] shall be decoded, which enables devices (indicated by the mask) to skip over irrelevant parts and only decode relevant IIF Blocks (of certain block\_ids). Furthermore, the signals being routed to specific devices are subject to further treatmeant in iif\_device\_specific\_routing\_config(). The information on the handling of audio signal roles assignment (e.g., for the echo reference signaling) and gain/delay parameters adjustment (e.g., to handle dynamic interactions between the listener and devices position), is decoded in this section of the IIF config. Another set of configuration parameters as specified in iif\_block\_stream\_tags(), enables the tagging of IIF Blocks for various purposes.

Finally, an IIF decoder shall decode information on the latency (max\_interleaving\_depth) and the number of predefined configuration updates (n\_config\_updates) specified by their indices (predef\_config\_upd\_idx). A configuration update shall be signaled via iif\_config\_update(\_cfg).

During the decoding of an IIF Block as specified in a later section, an IIF Block allows an inband signaling mechanism to carry a new set of configuration update utilizing the config index \_cfg of value 0. Apart from that, configuration updates can only be performed by decoding a particular config index \_cfg > 0 (dynamic\_config\_upd\_idx), chosen from a set of predefined config indices specified by iif\_specific\_config().

Configuration updates shall apply to the current frame only, i.e., updates shall not persist between frames. For each frame, the decoder shall reset to the initial parameters transmitted in iif\_specific\_config() before applying any configuration updates.

For AUDIO\_PRESENTATION\_TYPE\_CHANNELS, both channel superset and any channel subset may be selected during runtime. If an output interface is available, the decoder shall report the channel configuration change via the interface.

If the audio signal role for device d is updated (b\_override\_default\_role), the renderer shall adapt the signal routing from the IIF Audio Bus to the device according to the flag b\_signal\_connected; if set (1), the signal shall be routed to device d, otherwise the signal shall not be routed to device d. If available, the decoder shall report the role change for device d on the output interface.

Gain updates (b\_override\_default\_gain) shall be applied in the decoder based on the value iif\_apply\_gains. If the input interface is not available, gains shall be applied in the decoder. If available, the decoder shall report gain updates on the output interface.

Delay updates (b\_override\_default\_delay) shall be applied in the decoder based on the value iif\_apply\_delay. If the input interface is not available, delays shall always be applied in the decoder. If available, the decoder shall report delay updates on the output interface.

* + - * 1. **Decoding of IIF Block**

At least one IIF Block shall be decoded from an iif\_access\_unit() provided that the block is addressed to the decoder. Considering the latency info specified by max\_interleaving\_depth, the presence of redundancy blocks to cope with packet loss shall be verified. If no packet loss is detected, the extracted blocks shall be arranged based on the extracted information such as frame\_counter and block\_id. In the presence of multiple blocks having different block priority values, the decoder shall select the block with the lowest block priority.

An IIF Block is divided into three parts, i.e., header, payload and footer. The header shall decode information such as frame\_counter (indicating the corresponding audio frame of the block), block\_id, block\_priority, block\_size, b\_block\_protected (indicating the presence of CRC stored in the footer), reserved and block size extension (indicating a block\_size greater than 2047 bytes).

The iif\_block\_payload() shall decode information regarding the config update, metadata and audio data. For the config update, this shall also follow the mechanism described previously for iif\_specific\_config(). Parameters related to the predefined audio presentation layout of type channels (AUDIO\_PRESENTATION\_TYPE\_CHANNELS), can be updated. Additionally, when num\_target\_devices > 0, the dynamic parameters such as the audio signal roles configuration, gains and delays, can be updated as well. A relative delay (rel\_delay\_code) may be used to signal the delay relative to the default delay.

Furthermore, the presence of metadata sections in an IIF block shall be verified in iif\_metadata\_payload(). Specific metadata config indices and the corresponding metadata payload are decoded. An example of such metadata is the media authentication specified in auth\_extension(). This mechanism allows to carry authentication information where an authentication verification of protected messages shall be conducted to ensure their authenticity. The use of an explicit metadata signaling as specified in iif\_specific\_config() is to provide an alternative way of signaling the metadata config indices.

Next, iif\_block\_payload() shall verify the presence of an audio payload for the specified block\_id, and act accordingly. An audio payload consists of one or more channel elements of type Single Channel Element (SCE) or Channel Pair Element (CPE).

* + - * 1. **Decoder behaviour for tag and zone-based signal selection**

In case the iif\_device\_id is not set, a decoder shall, in a first step apply, the zone based filtering according to zone description (tag\_value equals to 14 ..16 in Table 4.93). The zone(s) of the device can be set by the iif\_tag\_id\_control\_value(s) of the iif\_tag\_id\_control equals to 15. A user presence can be set via iif\_tag\_id\_control equals to 14 with iif\_tag\_id\_control\_value equal to ‘1’ (user present). To inform the decoder about the presence of a specific user, an iif\_tag\_id\_value with the user number, e.g. 1, for the iif\_tag\_id\_control equals to 16 shall be signalled at the iif\_decoder\_interface.

For all remaining block streams selected for decoding/playout, the following metadata tag based signal selection shall be applied.

By default, the tag-based decision (skip\_state) whether a block shall be decoded or skipped is determined by the value b\_default\_skip as signaled in the bitstream.

The tag-based decision shall only be applied to blocks that are tagged, i.e. which have b\_block\_tags\_present set to 1.

The default decision can be updated by use of the selection interface which is processed as follows:

|  |
| --- |
| for i=1:num\_tag\_id\_controls: |
| if iif\_tag\_id\_control in [14..16]: |
| continue; |
| for all blocks in the stream with b\_block\_tags\_present==1: |
| for each block with block\_tag\_id that matches iif\_tag\_id\_control: |
| if b\_tag\_allows\_control==1 for this block\_tag\_id for this block: |
| if block has no block\_tag\_value or has block\_tag\_value==iif\_tag\_id\_control\_value for the currently evaluated block\_tag\_id: |
| if b\_mutually\_exclusive\_playout==0 for this block\_tag\_id: |
| skip\_state(selected block) = iif\_skip\_tag\_id\_no\_yes |
| else |
| if iif\_skip\_tag\_id\_no\_yes==0: |
| skip\_state(selected block) = 0 |
| skip\_state(all other blocks that have: block\_tag\_id matching iif\_tag\_id\_control, but have other tag value) = 1 |

The final decoder state regarding tag-based signal selection is given by the skip state after applying the above steps.

* + - * 1. **Decoder interfaces**

The last part of iif\_block\_payload decoding is to retrieve information on the presence of metadata, which can either be signaled explicitly or implicitly. If present, depending on the amount of metadata sections carried in the payload, it is then verified whether the corresponding metadata size is derived from the remaining payload bytes or specified in the payload (md\_size()). The metadata shall then specify the content based on the specified metadata type, such as METADATA\_TYPE\_AUTH for media authenticity metadata. In case the metadata\_type is “Reserved for use outside of ISO scope”, the IIF decoder shall convey the binary data to the calling layer.

**Table 4.XXY — Immersive interchange format decoder interface**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bits** | **Mnemonic** |
| iif\_decoder\_interface() |  |  |
| { |  |  |
| if (**iif\_input\_interface\_present**) | **1** | **uimsbf** |
| if (**iif\_device\_id\_present**) { | **1** | **uimsbf** |
| **iif\_device\_id**; | **8** | **uimsbf** |
| } |  |  |
| if (**iif\_gain\_control\_present**) { | **1** | **uimsbf** |
| **iif\_signal\_index**; | **8** | **uimsbf** |
| **iif\_apply\_gains**; | **1** | **uimsbf** |
| } |  |  |
| if (**iif\_delay\_control\_present**) { | **1** | **uimsbf** |
| **iif\_signal\_index**; | **8** | **uimsbf** |
| **iif\_apply\_delay**; | **1** | **uimsbf** |
| } |  |  |
| if (**iif\_tags\_present**) { | **1** | **uimsbf** |
| for(m = 0; m < **iif\_num\_tag\_id\_controls**; ++m) { | **4** | **uimsbf** |
| **iif\_tag\_id\_control;** | **8** | **uimsbf** |
| if (**block\_tag\_size**) { | **4** | **uimsbf** |
| **iif\_tag\_id\_control\_value**; | **8\*block\_tag\_size** | **uimsbf** |
| } |  |  |
| **iif\_skip\_tag\_id\_no\_yes**; | **1** | **uimsbf** |
| } |  |  |
| } |  |  |
| } else if (**iif\_output\_interface\_present**) { | **1** | **uimsbf** |
| if(**iif\_channel\_config\_changed**) { | **1** | **uimsbf** |
| **iif\_channel\_config\_mask**; | **32** | **uimsbf** |
| } |  |  |
| if(**iif\_audio\_signal\_role\_changed**) { | **1** | **uimsbf** |
| for(s = 0; s < **iif\_num\_signals\_device\_routing**; ++s) { | **8** | **uimsbf** |
| if(**iif\_audio\_signal\_role\_present**) { | **1** | **uimsbf** |
| **iif\_audio\_signal\_role;** | **8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |
| if(**iif\_gain\_changed**) { | **1** | **uimsbf** |
| for(s = 0; s < **iif\_num\_signals\_device\_routing**; ++s) { | **8** | **uimsbf** |
| if(**iif\_gain\_present**) { | **1** | **uimsbf** |
| **iif\_gain\_code;** | **8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |
| if(**iif\_delay\_changed**) { | **1** | **uimsbf** |
| for(s = 0; s < **iif\_num\_signals\_device\_routing**; ++s) { | **8** | **uimsbf** |
| if(**iif\_delay\_present**) { | **1** | **uimsbf** |
| **iif\_delay\_code;** | **8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |
| if (**iif\_block\_tag\_changed**) { | **1** | **uimsbf** |
| for(blk = 0; blk < **num\_block\_ids**; ++blk) { | **8** | **uimsbf** |
| for (i = 0; i < **iif\_num\_block\_tag\_ids**; ++i) { | **4** | **uimsbf** |
| **iif\_block\_tag\_id**; | **8** | **uimsbf** |
| for (s=0; s<**iif\_block\_tag\_size**; ++s) { | **4** | **uimsbf** |
| iif\_block\_tag\_value[s] = **iif\_block\_tag\_value**; | **8** | **uimsbf** |
| } |  |  |
| if (iif\_block\_tag\_id == 14 || iif\_block\_tag\_id ==16) { |  |  |
| for (e=0; e<**iif\_num\_excluded\_zones**; ++e) { | **4** | **uimsbf** |
| iif\_exluded\_zone[e] = **iif\_excluded\_zone**;\_ | **8** | **uimsbf** |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |
| } |  |  |

* + - * 1. **Semantics of iif\_decoder\_interface()**

|  |  |
| --- | --- |
| **iif\_input\_interface\_present** | Indicates whether the input interface is selected. |
| **iif\_device\_id\_present** | Indicates whether a device ID is present. |
| **iif\_device\_id** | The IIF device ID, 0.. **num\_target\_devices**-1. Device specific routing and zone tagging shall be mutually exclusive. When iif\_device\_id is provided, then the zone tagging shall be discarded. |
| **iif\_gain\_control\_present** | Indicates whether gain control information is present. |
| **iif\_signal\_index** | The index of the signal under device routing. |
| **iif\_apply\_gains** | Gains shall be applied in the decoder if set (1), else gains shall not be applied in the decoder. |
| **iif\_delay\_control\_present** | Indicates whether delay control information is present. |
| **iif\_apply\_delay** | Delay shall be applied in the decoder if set (1), else gains shall not be applied in the decoder. |
| **iif\_tags\_present** | Indicates whether tag control information is present. |
| **iif\_num\_tag\_id\_controls** | The number of control definitions that are used to determine which blocks shall be decoded or skipped. |
| **iif\_tag\_id\_control** | Used to control blocks tagged with block\_tag\_id == iif\_tag\_id\_control. iif\_tag\_id\_control==0 is interpreted as matching any block\_tag\_id signaled in the bitstream. If a tag has a value and b\_mutually\_exclusive\_playout == 0 for this tag, then iif\_tag\_id\_control==0 is also considered matching all block\_tag\_values. |
| **iif\_tag\_id\_control\_value** | Used to signal the control blocks tagged with block\_tag\_value==iif\_tag\_id\_control\_value. |
| **iif\_skip\_tag\_id\_no\_yes**; | Signals whether a block with matching block\_tag\_id (and block\_tag\_value, if applicable) is supposed to be decoded or skipped. |
| **iif\_output\_interface\_present** | Indicates whether the output interface is selected. |
| **iif\_channel\_config\_changed** | Indicates whether the channel configuration has changed. |
| **iif\_channel\_config\_mask** | The channel configuration mask, see **channel\_config\_set\_channel\_mode** |
| **iif\_audio\_signal\_role\_changed** | Indicates whether an audio signal role has changed. |
| **iif\_audio\_signal\_role** | The audio signal role, according to Table 4.ASR – Values of the audio\_signal\_role field. |
| **iif\_num\_signals\_device\_routing** | The number of signals assigned to device routing. |
| **iif\_gain\_changed** | Indicates whether an audio signal gain has changed. |
| **iif\_gain\_present** | Indicates whether an audio signal gain is present. |
| **iif\_gain\_code** | The gain code, see **gain\_code**. |
| **iif\_delay\_changed** | Indicates whether an audio signal delay has changed. |
| **iif\_delay\_present** | Indicates whether an audio signal delay is present. |
| **iif\_delay\_code** | The delay code, see **delay\_code**. |
| **iif\_block\_tag\_changed** | Indicates whether the block tag attribute of an audio signal has changed. |
| **iif\_num\_block\_tag\_ids** | The number of block tag IDs associated to the current signal. |
| **iif\_block\_tag\_id** | The ID of the IIF Block tag as defined in Table 4.93. |
| i**if\_block\_tag\_size** | The size of the iif\_block\_tag\_value for the current iif\_block\_tag\_id. |
| **iif\_block\_tag\_value** | The value associated to the current IIF Block tag ID as defined in Table 4.93. |
| **iif\_num\_excluded\_zones** | Signals the number of excluded zones, see **num\_excluded\_zones**. |
| **iif\_excluded\_zone** | Indicates an excluded zone for the zone based processing as defined in Table 4.93 and in the iif\_block\_stream\_zones(). |

Add the following Tables to the end of “4.A.2 Window tables”:

**Table 4.A.92 – Window coefficients wLDA for asymmetric low delay filterbank for N=240**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **n** | **wLDA(n)** | **n** | **wLDA(n)** | **n** | **wLDA(n)** |
| 0 | 0.000000000000000e+00 | 160 | 1.004708677491086e+00 | 320 | 9.964015284765968e-01 |
| 1 | 0.000000000000000e+00 | 161 | 1.005712337656749e+00 | 321 | 9.975915283480670e-01 |
| 2 | 0.000000000000000e+00 | 162 | 1.006627741823389e+00 | 322 | 9.988898206257559e-01 |
| 3 | 0.000000000000000e+00 | 163 | 1.007460860286395e+00 | 323 | 1.000303927234380e+00 |
| 4 | 0.000000000000000e+00 | 164 | 1.008215923600973e+00 | 324 | 1.001840787319378e+00 |
| 5 | 0.000000000000000e+00 | 165 | 1.008898689394160e+00 | 325 | 1.003508162416449e+00 |
| 6 | 0.000000000000000e+00 | 166 | 1.009512438049510e+00 | 326 | 1.005313192386866e+00 |
| 7 | 0.000000000000000e+00 | 167 | 1.010062133151922e+00 | 327 | 1.007263182132894e+00 |
| 8 | 0.000000000000000e+00 | 168 | 1.010550715971368e+00 | 328 | 1.009366925499550e+00 |
| 9 | 0.000000000000000e+00 | 169 | 1.010982135506986e+00 | 329 | 1.011654901040475e+00 |
| 10 | 0.000000000000000e+00 | 170 | 1.011359143572147e+00 | 330 | 1.013460916839174e+00 |
| 11 | 0.000000000000000e+00 | 171 | 1.011685173724601e+00 | 331 | 1.015439859549357e+00 |
| 12 | 0.000000000000000e+00 | 172 | 1.011962331100864e+00 | 332 | 1.017510118327508e+00 |
| 13 | 0.000000000000000e+00 | 173 | 1.012194068117524e+00 | 333 | 1.019521566634239e+00 |
| 14 | 0.000000000000000e+00 | 174 | 1.012382309473470e+00 | 334 | 1.021427210603284e+00 |
| 15 | 0.000000000000000e+00 | 175 | 1.012530134411619e+00 | 335 | 1.023175976163407e+00 |
| 16 | 0.000000000000000e+00 | 176 | 1.012639775003457e+00 | 336 | 1.024716149969084e+00 |
| 17 | 0.000000000000000e+00 | 177 | 1.012713798678211e+00 | 337 | 1.025991493983836e+00 |
| 18 | 0.000000000000000e+00 | 178 | 1.012755013843985e+00 | 338 | 1.026942795115598e+00 |
| 19 | 0.000000000000000e+00 | 179 | 1.012765799894453e+00 | 339 | 1.027511063644843e+00 |
| 20 | 0.000000000000000e+00 | 180 | 1.012748952907404e+00 | 340 | 1.027634294456205e+00 |
| 21 | 0.000000000000000e+00 | 181 | 1.012706955800289e+00 | 341 | 1.027250978292214e+00 |
| 22 | 0.000000000000000e+00 | 182 | 1.012642857380847e+00 | 342 | 1.026304095700693e+00 |
| 23 | 0.000000000000000e+00 | 183 | 1.012558879696851e+00 | 343 | 1.024736164069697e+00 |
| 24 | 0.000000000000000e+00 | 184 | 1.012458183122033e+00 | 344 | 1.022490937034641e+00 |
| 25 | 0.000000000000000e+00 | 185 | 1.012342907951334e+00 | 345 | 1.019515072412387e+00 |
| 26 | 0.000000000000000e+00 | 186 | 1.012216235487353e+00 | 346 | 1.015760373791603e+00 |
| 27 | 0.000000000000000e+00 | 187 | 1.012080125934687e+00 | 347 | 1.011183971347815e+00 |
| 28 | 0.000000000000000e+00 | 188 | 1.011937567254485e+00 | 348 | 1.005746084650236e+00 |
| 29 | 0.000000000000000e+00 | 189 | 1.011790282474963e+00 | 349 | 9.994114730415857e-01 |
| 30 | 0.000000000000000e+00 | 190 | 1.011641272406023e+00 | 350 | 9.921529097154242e-01 |
| 31 | 0.000000000000000e+00 | 191 | 1.011491904228184e+00 | 351 | 9.839435385219192e-01 |
| 32 | 0.000000000000000e+00 | 192 | 1.011344700694417e+00 | 352 | 9.747634827941575e-01 |
| 33 | 0.000000000000000e+00 | 193 | 1.011201071127927e+00 | 353 | 9.646048250501910e-01 |
| 34 | 0.000000000000000e+00 | 194 | 1.011062876503041e+00 | 354 | 9.534619388400459e-01 |
| 35 | 0.000000000000000e+00 | 195 | 1.010931436739342e+00 | 355 | 9.413348145272842e-01 |
| 36 | 0.000000000000000e+00 | 196 | 1.010808068774316e+00 | 356 | 9.282282546151819e-01 |
| 37 | 0.000000000000000e+00 | 197 | 1.010693501186928e+00 | 357 | 9.141540563656075e-01 |
| 38 | 0.000000000000000e+00 | 198 | 1.010588873699422e+00 | 358 | 8.991320235484349e-01 |
| 39 | 0.000000000000000e+00 | 199 | 1.010494354532353e+00 | 359 | 8.831896853053627e-01 |
| 40 | 0.000000000000000e+00 | 200 | 1.010410221483215e+00 | 360 | 8.663548164329093e-01 |
| 41 | 0.000000000000000e+00 | 201 | 1.010336490294771e+00 | 361 | 8.486643364959733e-01 |
| 42 | 0.000000000000000e+00 | 202 | 1.010272524848519e+00 | 362 | 8.301579139160681e-01 |
| 43 | 0.000000000000000e+00 | 203 | 1.010218096148412e+00 | 363 | 8.108788692151807e-01 |
| 44 | 0.000000000000000e+00 | 204 | 1.010171656775640e+00 | 364 | 7.908752989295335e-01 |
| 45 | 0.000000000000000e+00 | 205 | 1.010132323996401e+00 | 365 | 7.701990187606995e-01 |
| 46 | 0.000000000000000e+00 | 206 | 1.010098388689342e+00 | 366 | 7.489068963384272e-01 |
| 47 | 0.000000000000000e+00 | 207 | 1.010068202038694e+00 | 367 | 7.270576699841359e-01 |
| 48 | 0.000000000000000e+00 | 208 | 1.010039453539107e+00 | 368 | 7.047092819314779e-01 |
| 49 | 0.000000000000000e+00 | 209 | 1.010010230290263e+00 | 369 | 6.819230974423550e-01 |
| 50 | 0.000000000000000e+00 | 210 | 1.009977916643680e+00 | 370 | 6.587619767809000e-01 |
| 51 | 0.000000000000000e+00 | 211 | 1.009940336228694e+00 | 371 | 6.352890592874099e-01 |
| 52 | 0.000000000000000e+00 | 212 | 1.009894548257807e+00 | 372 | 6.115695310143157e-01 |
| 53 | 0.000000000000000e+00 | 213 | 1.009838308708799e+00 | 373 | 5.876661722564289e-01 |
| 54 | 0.000000000000000e+00 | 214 | 1.009768980817384e+00 | 374 | 5.636399817032525e-01 |
| 55 | 0.000000000000000e+00 | 215 | 1.009684090687164e+00 | 375 | 5.395557644293222e-01 |
| 56 | 0.000000000000000e+00 | 216 | 1.009581280555682e+00 | 376 | 5.154735456539700e-01 |
| 57 | 0.000000000000000e+00 | 217 | 1.009458241425143e+00 | 377 | 4.914498631045615e-01 |
| 58 | 0.000000000000000e+00 | 218 | 1.009313224929575e+00 | 378 | 4.675442291623012e-01 |
| 59 | 0.000000000000000e+00 | 219 | 1.009144112734761e+00 | 379 | 4.438114799213576e-01 |
| 60 | 0.000000000000000e+00 | 220 | 1.008949493525753e+00 | 380 | 4.203043046885219e-01 |
| 61 | 0.000000000000000e+00 | 221 | 1.008727884997619e+00 | 381 | 3.970739591211551e-01 |
| 62 | 0.000000000000000e+00 | 222 | 1.008478423284851e+00 | 382 | 3.741642373060188e-01 |
| 63 | 0.000000000000000e+00 | 223 | 1.008200146369426e+00 | 383 | 3.516171481750350e-01 |
| 64 | 0.000000000000000e+00 | 224 | 1.007892674961336e+00 | 384 | 3.294769437072290e-01 |
| 65 | 0.000000000000000e+00 | 225 | 1.007555573455895e+00 | 385 | 3.077789078889820e-01 |
| 66 | 0.000000000000000e+00 | 226 | 1.007189345025525e+00 | 386 | 2.865543814049772e-01 |
| 67 | 0.000000000000000e+00 | 227 | 1.006793927824538e+00 | 387 | 2.658346019332584e-01 |
| 68 | 0.000000000000000e+00 | 228 | 1.006370303149314e+00 | 388 | 2.456456803467095e-01 |
| 69 | 0.000000000000000e+00 | 229 | 1.005919224127911e+00 | 389 | 2.260093000067393e-01 |
| 70 | 0.000000000000000e+00 | 230 | 1.005442141810160e+00 | 390 | 2.069449962574092e-01 |
| 71 | 0.000000000000000e+00 | 231 | 1.004940548815171e+00 | 391 | 1.884686389218322e-01 |
| 72 | 0.000000000000000e+00 | 232 | 1.004416038098237e+00 | 392 | 1.705954709184399e-01 |
| 73 | 0.000000000000000e+00 | 233 | 1.003870934089686e+00 | 393 | 1.533343890681390e-01 |
| 74 | 0.000000000000000e+00 | 234 | 1.003307449770187e+00 | 394 | 1.366911019414417e-01 |
| 75 | 0.000000000000000e+00 | 235 | 1.002728111386909e+00 | 395 | 1.206703467513333e-01 |
| 76 | 0.000000000000000e+00 | 236 | 1.002135464655177e+00 | 396 | 1.052747118478660e-01 |
| 77 | 0.000000000000000e+00 | 237 | 1.001532636590676e+00 | 397 | 9.050369257152079e-02 |
| 78 | 0.000000000000000e+00 | 238 | 1.000922365901945e+00 | 398 | 7.635397728566121e-02 |
| 79 | 0.000000000000000e+00 | 239 | 1.000308193833526e+00 | 399 | 6.282034030592902e-02 |
| 80 | 0.000000000000000e+00 | 240 | 9.996919011206490e-01 | 400 | 4.989736509080558e-02 |
| 81 | 0.000000000000000e+00 | 241 | 9.990784840729244e-01 | 401 | 3.757652772246801e-02 |
| 82 | 0.000000000000000e+00 | 242 | 9.984697087896268e-01 | 402 | 2.584738191459814e-02 |
| 83 | 0.000000000000000e+00 | 243 | 9.978690858367024e-01 | 403 | 1.470155068746303e-02 |
| 84 | 0.000000000000000e+00 | 244 | 9.972793109558521e-01 | 404 | 4.126595858583295e-03 |
| 85 | 0.000000000000000e+00 | 245 | 9.967034533921340e-01 | 405 | -5.891434269890659e-03 |
| 86 | 0.000000000000000e+00 | 246 | 9.961439922621166e-01 | 406 | -1.536329520788766e-02 |
| 87 | 0.000000000000000e+00 | 247 | 9.956033775539884e-01 | 407 | -2.430122582451853e-02 |
| 88 | 0.000000000000000e+00 | 248 | 9.950837402063278e-01 | 408 | -3.272045590229335e-02 |
| 89 | 0.000000000000000e+00 | 249 | 9.945873147903244e-01 | 409 | -4.063362855701623e-02 |
| 90 | 5.305235098871444e-03 | 250 | 9.941156069136238e-01 | 410 | -4.805444424454820e-02 |
| 91 | 9.996743588367531e-03 | 251 | 9.936700207375173e-01 | 411 | -5.499497258200362e-02 |
| 92 | 1.460796755137538e-02 | 252 | 9.932519181564613e-01 | 412 | -6.146688124166948e-02 |
| 93 | 2.024370180670145e-02 | 253 | 9.928619727154252e-01 | 413 | -6.748468182105991e-02 |
| 94 | 2.701461405056267e-02 | 254 | 9.925010851461232e-01 | 414 | -7.306142427456862e-02 |
| 95 | 3.499361000717621e-02 | 255 | 9.921691315380984e-01 | 415 | -7.820617483254730e-02 |
| 96 | 4.424588851571281e-02 | 256 | 9.918665491182922e-01 | 416 | -8.293043914044632e-02 |
| 97 | 5.482436608328485e-02 | 257 | 9.915928560402563e-01 | 417 | -8.724644532866996e-02 |
| 98 | 6.676765847398403e-02 | 258 | 9.913476318763218e-01 | 418 | -9.116452506492527e-02 |
| 99 | 8.009914366829558e-02 | 259 | 9.911298894709990e-01 | 419 | -9.469304216883027e-02 |
| 100 | 9.482665349502306e-02 | 260 | 9.909387444078919e-01 | 420 | -9.784120023411771e-02 |
| 101 | 1.109422548710344e-01 | 261 | 9.907727108894024e-01 | 421 | -1.006190418791648e-01 |
| 102 | 1.284223074167396e-01 | 262 | 9.906303787150326e-01 | 422 | -1.030360111111103e-01 |
| 103 | 1.472287968327706e-01 | 263 | 9.905096491583045e-01 | 423 | -1.050995803285455e-01 |
| 104 | 1.673100807904834e-01 | 264 | 9.904087914462694e-01 | 424 | -1.068172142230882e-01 |
| 105 | 1.885997642296488e-01 | 265 | 9.903255289051605e-01 | 425 | -1.081974227416502e-01 |
| 106 | 2.110209448747974e-01 | 266 | 9.902575406142213e-01 | 426 | -1.092492774392824e-01 |
| 107 | 2.344880603710975e-01 | 267 | 9.902023946214220e-01 | 427 | -1.099809851424550e-01 |
| 108 | 2.589033711808454e-01 | 268 | 9.901575015155720e-01 | 428 | -1.103977408741932e-01 |
| 109 | 2.841647033392418e-01 | 269 | 9.901206586012907e-01 | 429 | -1.105084619148789e-01 |
| 110 | 3.101627843160973e-01 | 270 | 9.900889812894406e-01 | 430 | -1.103242262600913e-01 |
| 111 | 3.367837772954476e-01 | 271 | 9.900603352632121e-01 | 431 | -1.098524491515805e-01 |
| 112 | 3.639114681156252e-01 | 272 | 9.900321562263099e-01 | 432 | -1.090997300590506e-01 |
| 113 | 3.914252910998820e-01 | 273 | 9.900025692522435e-01 | 433 | -1.080766457616878e-01 |
| 114 | 4.192049917945288e-01 | 274 | 9.899693102025343e-01 | 434 | -1.067939984687745e-01 |
| 115 | 4.471309850999535e-01 | 275 | 9.899307640365727e-01 | 435 | -1.052637003544443e-01 |
| 116 | 4.750868825511614e-01 | 276 | 9.898852572653667e-01 | 436 | -1.034961241263523e-01 |
| 117 | 5.029523957059122e-01 | 277 | 9.898319269347060e-01 | 437 | -1.015008467688577e-01 |
| 118 | 5.306181649316717e-01 | 278 | 9.897692596535289e-01 | 438 | -9.929123258537813e-02 |
| 119 | 5.579747428663487e-01 | 279 | 9.896970346678272e-01 | 439 | -9.688303623049198e-02 |
| 120 | 5.849206604934815e-01 | 280 | 9.896146331889107e-01 | 440 | -9.428847446755965e-02 |
| 121 | 6.113488038789190e-01 | 281 | 9.895220757174378e-01 | 441 | -9.152091100798199e-02 |
| 122 | 6.371684119604742e-01 | 282 | 9.894196399062921e-01 | 442 | -8.859705468085925e-02 |
| 123 | 6.622962432368731e-01 | 283 | 9.893074965384659e-01 | 443 | -8.553324579905185e-02 |
| 124 | 6.866580716267418e-01 | 284 | 9.891867674284610e-01 | 444 | -8.234442557322251e-02 |
| 125 | 7.101790843209148e-01 | 285 | 9.890581715933395e-01 | 445 | -7.904647440788692e-02 |
| 126 | 7.327963552921717e-01 | 286 | 9.889230031022089e-01 | 446 | -7.565695975592170e-02 |
| 127 | 7.544551974836834e-01 | 287 | 9.887825578295630e-01 | 447 | -7.219447805250771e-02 |
| 128 | 7.751108841891807e-01 | 288 | 9.886386592120545e-01 | 448 | -6.867606753531441e-02 |
| 129 | 7.947291447585267e-01 | 289 | 9.884926873551375e-01 | 449 | -6.511708163113709e-02 |
| 130 | 8.132817365236141e-01 | 290 | 9.883471084085503e-01 | 450 | -6.153426201732499e-02 |
| 131 | 8.307479727020245e-01 | 291 | 9.882032571565917e-01 | 451 | -5.794654834034055e-02 |
| 132 | 8.471200801854385e-01 | 292 | 9.880640617030884e-01 | 452 | -5.437166635159518e-02 |
| 133 | 8.623984077953557e-01 | 293 | 9.879311998177238e-01 | 453 | -5.082325342672667e-02 |
| 134 | 8.765908974996186e-01 | 294 | 9.878075819424549e-01 | 454 | -4.731768261606175e-02 |
| 135 | 8.897163275605041e-01 | 295 | 9.876951134084213e-01 | 455 | -4.387219218706189e-02 |
| 136 | 9.018003682081348e-01 | 296 | 9.875968894760857e-01 | 456 | -4.050053247568393e-02 |
| 137 | 9.128745354570823e-01 | 297 | 9.875149888347144e-01 | 457 | -3.721502082245055e-02 |
| 138 | 9.229775478442888e-01 | 298 | 9.874524849192456e-01 | 458 | -3.402996266948349e-02 |
| 139 | 9.321553861564006e-01 | 299 | 9.874115368168944e-01 | 459 | -3.095765092956728e-02 |
| 140 | 9.404609052933396e-01 | 300 | 9.873951115886979e-01 | 460 | -2.800909464274782e-02 |
| 141 | 9.479479345282376e-01 | 301 | 9.874056275509585e-01 | 461 | -2.519406300389030e-02 |
| 142 | 9.546732976073706e-01 | 302 | 9.874458127312921e-01 | 462 | -2.252080561390149e-02 |
| 143 | 9.606976399184645e-01 | 303 | 9.875179947346887e-01 | 463 | -1.999721557401502e-02 |
| 144 | 9.660805524695870e-01 | 304 | 9.876249269174586e-01 | 464 | -1.762979910961727e-02 |
| 145 | 9.708821747608998e-01 | 305 | 9.877691368590689e-01 | 465 | -1.542181679511618e-02 |
| 146 | 9.751623811486372e-01 | 306 | 9.879528358230726e-01 | 466 | -1.337713096257934e-02 |
| 147 | 9.789747333404933e-01 | 307 | 9.881790747212391e-01 | 467 | -1.149880303071551e-02 |
| 148 | 9.823765865983286e-01 | 308 | 9.884497924570929e-01 | 468 | -9.785314755557023e-03 |
| 149 | 9.855347660016696e-01 | 309 | 9.887684373604154e-01 | 469 | -8.233976327300612e-03 |
| 150 | 9.884793707533194e-01 | 310 | 9.891371616557014e-01 | 470 | -6.843234536105624e-03 |
| 151 | 9.907199995730845e-01 | 311 | 9.895594394179152e-01 | 471 | -5.609327243712055e-03 |
| 152 | 9.927891912841345e-01 | 312 | 9.900381047643838e-01 | 472 | -4.525198076002370e-03 |
| 153 | 9.947148884277037e-01 | 313 | 9.905771957917731e-01 | 473 | -3.584129920673041e-03 |
| 154 | 9.965041017623596e-01 | 314 | 9.911797988363887e-01 | 474 | -2.778420871815740e-03 |
| 155 | 9.981625949525345e-01 | 315 | 9.918510277326026e-01 | 475 | -2.098197727900724e-03 |
| 156 | 9.996961651093181e-01 | 316 | 9.925943919208190e-01 | 476 | -1.533478537964509e-03 |
| 157 | 1.001111413242302e+00 | 317 | 9.934158959186011e-01 | 477 | -1.074443637110903e-03 |
| 158 | 1.002414286392268e+00 | 318 | 9.943201078053212e-01 | 478 | -7.078546706512391e-04 |
| 159 | 1.003611467285588e+00 | 319 | 9.953133902427869e-01 | 479 | -3.613496418928369e-04 |

**Table 4.A.93 – Window coefficients wLDA for asymmetric low delay filterbank for N=480**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **n** | **wLDA(n)** | **n** | **wLDA(n)** | **n** | **wLDA(n)** |
| 0 | 0.000000000000000e+00 | 320 | 1.004443531995945e+00 | 640 | 9.961203379971326e-01 |
| 1 | 0.000000000000000e+00 | 321 | 1.004967986424467e+00 | 641 | 9.966890948441745e-01 |
| 2 | 0.000000000000000e+00 | 322 | 1.005470005637052e+00 | 642 | 9.972842679758880e-01 |
| 3 | 0.000000000000000e+00 | 323 | 1.005949145170711e+00 | 643 | 9.979055415627330e-01 |
| 4 | 0.000000000000000e+00 | 324 | 1.006407009542103e+00 | 644 | 9.985548127155425e-01 |
| 5 | 0.000000000000000e+00 | 325 | 1.006843352506855e+00 | 645 | 9.992320848298057e-01 |
| 6 | 0.000000000000000e+00 | 326 | 1.007260142622887e+00 | 646 | 9.999393169239009e-01 |
| 7 | 0.000000000000000e+00 | 327 | 1.007656661215973e+00 | 647 | 1.000676188436651e+00 |
| 8 | 0.000000000000000e+00 | 328 | 1.008034308295421e+00 | 648 | 1.001444733905817e+00 |
| 9 | 0.000000000000000e+00 | 329 | 1.008393079963091e+00 | 649 | 1.002245009135684e+00 |
| 10 | 0.000000000000000e+00 | 330 | 1.008734758578989e+00 | 650 | 1.003078846506546e+00 |
| 11 | 0.000000000000000e+00 | 331 | 1.009058249873833e+00 | 651 | 1.003946083413733e+00 |
| 12 | 0.000000000000000e+00 | 332 | 1.009365251564694e+00 | 652 | 1.004848753962734e+00 |
| 13 | 0.000000000000000e+00 | 333 | 1.009655660215271e+00 | 653 | 1.005786648810854e+00 |
| 14 | 0.000000000000000e+00 | 334 | 1.009930754807923e+00 | 654 | 1.006761700412993e+00 |
| 15 | 0.000000000000000e+00 | 335 | 1.010189681124657e+00 | 655 | 1.007773858455400e+00 |
| 16 | 0.000000000000000e+00 | 336 | 1.010434184200640e+00 | 656 | 1.008824888092573e+00 |
| 17 | 0.000000000000000e+00 | 337 | 1.010663676735228e+00 | 657 | 1.009914592130044e+00 |
| 18 | 0.000000000000000e+00 | 338 | 1.010879608424312e+00 | 658 | 1.011044869639164e+00 |
| 19 | 0.000000000000000e+00 | 339 | 1.011081255645969e+00 | 659 | 1.012216130365610e+00 |
| 20 | 0.000000000000000e+00 | 340 | 1.011269960947744e+00 | 660 | 1.013047565149327e+00 |
| 21 | 0.000000000000000e+00 | 341 | 1.011445162723270e+00 | 661 | 1.013915946100629e+00 |
| 22 | 0.000000000000000e+00 | 342 | 1.011608449127642e+00 | 662 | 1.014923441259795e+00 |
| 23 | 0.000000000000000e+00 | 343 | 1.011758810583150e+00 | 663 | 1.015957433206324e+00 |
| 24 | 0.000000000000000e+00 | 344 | 1.011897560567707e+00 | 664 | 1.016996499560845e+00 |
| 25 | 0.000000000000000e+00 | 345 | 1.012024302085441e+00 | 665 | 1.018019442784231e+00 |
| 26 | 0.000000000000000e+00 | 346 | 1.012140460211194e+00 | 666 | 1.019027285501251e+00 |
| 27 | 0.000000000000000e+00 | 347 | 1.012244921966297e+00 | 667 | 1.020009139549275e+00 |
| 28 | 0.000000000000000e+00 | 348 | 1.012339226241367e+00 | 668 | 1.020963871317665e+00 |
| 29 | 0.000000000000000e+00 | 349 | 1.012422888521601e+00 | 669 | 1.021880604350422e+00 |
| 30 | 0.000000000000000e+00 | 350 | 1.012497050269805e+00 | 670 | 1.022756514615106e+00 |
| 31 | 0.000000000000000e+00 | 351 | 1.012560833005713e+00 | 671 | 1.023582385618774e+00 |
| 32 | 0.000000000000000e+00 | 352 | 1.012615904116265e+00 | 672 | 1.024353980976701e+00 |
| 33 | 0.000000000000000e+00 | 353 | 1.012661400539405e+00 | 673 | 1.025061777648234e+00 |
| 34 | 0.000000000000000e+00 | 354 | 1.012698607692183e+00 | 674 | 1.025701002415063e+00 |
| 35 | 0.000000000000000e+00 | 355 | 1.012726958954961e+00 | 675 | 1.026261663878092e+00 |
| 36 | 0.000000000000000e+00 | 356 | 1.012747819936584e+00 | 676 | 1.026738673647482e+00 |
| 37 | 0.000000000000000e+00 | 357 | 1.012760298661069e+00 | 677 | 1.027122986826984e+00 |
| 38 | 0.000000000000000e+00 | 358 | 1.012765922449960e+00 | 678 | 1.027408519661012e+00 |
| 39 | 0.000000000000000e+00 | 359 | 1.012763948841204e+00 | 679 | 1.027585830688143e+00 |
| 40 | 0.000000000000000e+00 | 360 | 1.012755753887085e+00 | 680 | 1.027648951922701e+00 |
| 41 | 0.000000000000000e+00 | 361 | 1.012740575296603e+00 | 681 | 1.027587902203109e+00 |
| 42 | 0.000000000000000e+00 | 362 | 1.012719789201144e+00 | 682 | 1.027397523939210e+00 |
| 43 | 0.000000000000000e+00 | 363 | 1.012692758750213e+00 | 683 | 1.027069179375841e+00 |
| 44 | 0.000000000000000e+00 | 364 | 1.012660975529858e+00 | 684 | 1.026596938589443e+00 |
| 45 | 0.000000000000000e+00 | 365 | 1.012623472898504e+00 | 685 | 1.025972450969440e+00 |
| 46 | 0.000000000000000e+00 | 366 | 1.012581678169188e+00 | 686 | 1.025189721888128e+00 |
| 47 | 0.000000000000000e+00 | 367 | 1.012535058602453e+00 | 687 | 1.024240262467000e+00 |
| 48 | 0.000000000000000e+00 | 368 | 1.012484972246822e+00 | 688 | 1.023118841384460e+00 |
| 49 | 0.000000000000000e+00 | 369 | 1.012430463679316e+00 | 689 | 1.021817328911793e+00 |
| 50 | 0.000000000000000e+00 | 370 | 1.012373028737774e+00 | 690 | 1.020330464463111e+00 |
| 51 | 0.000000000000000e+00 | 371 | 1.012312095063070e+00 | 691 | 1.018650990561848e+00 |
| 52 | 0.000000000000000e+00 | 372 | 1.012249023976742e+00 | 692 | 1.016774659209400e+00 |
| 53 | 0.000000000000000e+00 | 373 | 1.012182837094955e+00 | 693 | 1.014694702432600e+00 |
| 54 | 0.000000000000000e+00 | 374 | 1.012114985644919e+00 | 694 | 1.012407428282884e+00 |
| 55 | 0.000000000000000e+00 | 375 | 1.012044885003304e+00 | 695 | 1.009906654565108e+00 |
| 56 | 0.000000000000000e+00 | 376 | 1.011973876511603e+00 | 696 | 1.007188578458507e+00 |
| 57 | 0.000000000000000e+00 | 377 | 1.011900936547881e+00 | 697 | 1.004247514102417e+00 |
| 58 | 0.000000000000000e+00 | 378 | 1.011827484797985e+00 | 698 | 1.001080961257010e+00 |
| 59 | 0.000000000000000e+00 | 379 | 1.011753008569803e+00 | 699 | 9.976842395284637e-01 |
| 60 | 0.000000000000000e+00 | 380 | 1.011678803938046e+00 | 700 | 9.940557180662704e-01 |
| 61 | 0.000000000000000e+00 | 381 | 1.011603680910505e+00 | 701 | 9.901906380163595e-01 |
| 62 | 0.000000000000000e+00 | 382 | 1.011529185153809e+00 | 702 | 9.860865871353246e-01 |
| 63 | 0.000000000000000e+00 | 383 | 1.011454785713102e+00 | 703 | 9.817397501303696e-01 |
| 64 | 0.000000000000000e+00 | 384 | 1.011381453638912e+00 | 704 | 9.771501187120180e-01 |
| 65 | 0.000000000000000e+00 | 385 | 1.011308167670753e+00 | 705 | 9.723156637834687e-01 |
| 66 | 0.000000000000000e+00 | 386 | 1.011236610341260e+00 | 706 | 9.672366712325431e-01 |
| 67 | 0.000000000000000e+00 | 387 | 1.011165861239173e+00 | 707 | 9.619114522936701e-01 |
| 68 | 0.000000000000000e+00 | 388 | 1.011096993993037e+00 | 708 | 9.563402921286364e-01 |
| 69 | 0.000000000000000e+00 | 389 | 1.011029191301694e+00 | 709 | 9.505220861927248e-01 |
| 70 | 0.000000000000000e+00 | 390 | 1.010963729626641e+00 | 710 | 9.444588390359354e-01 |
| 71 | 0.000000000000000e+00 | 391 | 1.010899655674578e+00 | 711 | 9.381494858921667e-01 |
| 72 | 0.000000000000000e+00 | 392 | 1.010838252644461e+00 | 712 | 9.315962496426278e-01 |
| 73 | 0.000000000000000e+00 | 393 | 1.010778416755264e+00 | 713 | 9.247997426966093e-01 |
| 74 | 0.000000000000000e+00 | 394 | 1.010721360243234e+00 | 714 | 9.177625550620636e-01 |
| 75 | 0.000000000000000e+00 | 395 | 1.010666285876455e+00 | 715 | 9.104863121445679e-01 |
| 76 | 0.000000000000000e+00 | 396 | 1.010614266894512e+00 | 716 | 9.029751680353524e-01 |
| 77 | 0.000000000000000e+00 | 397 | 1.010564099281000e+00 | 717 | 8.952313288619367e-01 |
| 78 | 0.000000000000000e+00 | 398 | 1.010517096381509e+00 | 718 | 8.872599706865123e-01 |
| 79 | 0.000000000000000e+00 | 399 | 1.010472266495074e+00 | 719 | 8.790631561061171e-01 |
| 80 | 0.000000000000000e+00 | 400 | 1.010430470494512e+00 | 720 | 8.706456337542150e-01 |
| 81 | 0.000000000000000e+00 | 401 | 1.010390635488037e+00 | 721 | 8.620108336276733e-01 |
| 82 | 0.000000000000000e+00 | 402 | 1.010354079663767e+00 | 722 | 8.531651194538425e-01 |
| 83 | 0.000000000000000e+00 | 403 | 1.010319484524512e+00 | 723 | 8.441125827416620e-01 |
| 84 | 0.000000000000000e+00 | 404 | 1.010287732968580e+00 | 724 | 8.348589373399948e-01 |
| 85 | 0.000000000000000e+00 | 405 | 1.010257950227988e+00 | 725 | 8.254086223972127e-01 |
| 86 | 0.000000000000000e+00 | 406 | 1.010231078494249e+00 | 726 | 8.157687070715176e-01 |
| 87 | 0.000000000000000e+00 | 407 | 1.010205587931660e+00 | 727 | 8.059437233154637e-01 |
| 88 | 0.000000000000000e+00 | 408 | 1.010182635897876e+00 | 728 | 7.959414651061612e-01 |
| 89 | 0.000000000000000e+00 | 409 | 1.010161131093372e+00 | 729 | 7.857670170752049e-01 |
| 90 | 0.000000000000000e+00 | 410 | 1.010141762173145e+00 | 730 | 7.754281892901473e-01 |
| 91 | 0.000000000000000e+00 | 411 | 1.010123226584120e+00 | 731 | 7.649313654540612e-01 |
| 92 | 0.000000000000000e+00 | 412 | 1.010106564965965e+00 | 732 | 7.542846327442048e-01 |
| 93 | 0.000000000000000e+00 | 413 | 1.010090449982098e+00 | 733 | 7.434943718765665e-01 |
| 94 | 0.000000000000000e+00 | 414 | 1.010075674445289e+00 | 734 | 7.325691772738999e-01 |
| 95 | 0.000000000000000e+00 | 415 | 1.010060809299530e+00 | 735 | 7.215149331458728e-01 |
| 96 | 0.000000000000000e+00 | 416 | 1.010046722045583e+00 | 736 | 7.103400549562944e-01 |
| 97 | 0.000000000000000e+00 | 417 | 1.010032170180360e+00 | 737 | 6.990511539119996e-01 |
| 98 | 0.000000000000000e+00 | 418 | 1.010017890428877e+00 | 738 | 6.876573952173626e-01 |
| 99 | 0.000000000000000e+00 | 419 | 1.010002363879044e+00 | 739 | 6.761653499550255e-01 |
| 100 | 0.000000000000000e+00 | 420 | 1.009986499185054e+00 | 740 | 6.645840786371451e-01 |
| 101 | 0.000000000000000e+00 | 421 | 1.009969021400474e+00 | 741 | 6.529203544876097e-01 |
| 102 | 0.000000000000000e+00 | 422 | 1.009950573483366e+00 | 742 | 6.411830842823245e-01 |
| 103 | 0.000000000000000e+00 | 423 | 1.009929567021562e+00 | 743 | 6.293796611336280e-01 |
| 104 | 0.000000000000000e+00 | 424 | 1.009906958448099e+00 | 744 | 6.175192060085208e-01 |
| 105 | 0.000000000000000e+00 | 425 | 1.009881498943010e+00 | 745 | 6.056083823929643e-01 |
| 106 | 0.000000000000000e+00 | 426 | 1.009853706282597e+00 | 746 | 5.936560624526468e-01 |
| 107 | 0.000000000000000e+00 | 427 | 1.009822090220407e+00 | 747 | 5.816685576268035e-01 |
| 108 | 0.000000000000000e+00 | 428 | 1.009787744284661e+00 | 748 | 5.696546730080154e-01 |
| 109 | 0.000000000000000e+00 | 429 | 1.009749238562067e+00 | 749 | 5.576217157959890e-01 |
| 110 | 0.000000000000000e+00 | 430 | 1.009707093778162e+00 | 750 | 5.455789811615239e-01 |
| 111 | 0.000000000000000e+00 | 431 | 1.009659973830751e+00 | 751 | 5.335326819631583e-01 |
| 112 | 0.000000000000000e+00 | 432 | 1.009608886895764e+00 | 752 | 5.214909841729947e-01 |
| 113 | 0.000000000000000e+00 | 433 | 1.009552401900961e+00 | 753 | 5.094597228333853e-01 |
| 114 | 0.000000000000000e+00 | 434 | 1.009491175244507e+00 | 754 | 4.974471592901086e-01 |
| 115 | 0.000000000000000e+00 | 435 | 1.009423944949143e+00 | 755 | 4.854600184866710e-01 |
| 116 | 0.000000000000000e+00 | 436 | 1.009351762546296e+00 | 756 | 4.735067030065239e-01 |
| 117 | 0.000000000000000e+00 | 437 | 1.009273163797313e+00 | 757 | 4.615925445090212e-01 |
| 118 | 0.000000000000000e+00 | 438 | 1.009188880897370e+00 | 758 | 4.497256320417501e-01 |
| 119 | 0.000000000000000e+00 | 439 | 1.009097763850333e+00 | 759 | 4.379113897565727e-01 |
| 120 | 0.000000000000000e+00 | 440 | 1.009000766336071e+00 | 760 | 4.261571642320424e-01 |
| 121 | 0.000000000000000e+00 | 441 | 1.008896526233555e+00 | 761 | 4.144688328137527e-01 |
| 122 | 0.000000000000000e+00 | 442 | 1.008786009787269e+00 | 762 | 4.028532873867052e-01 |
| 123 | 0.000000000000000e+00 | 443 | 1.008668018334797e+00 | 763 | 3.913146885756875e-01 |
| 124 | 0.000000000000000e+00 | 444 | 1.008543573152632e+00 | 764 | 3.798595119591716e-01 |
| 125 | 0.000000000000000e+00 | 445 | 1.008411468616852e+00 | 765 | 3.684915649120530e-01 |
| 126 | 0.000000000000000e+00 | 446 | 1.008272602383284e+00 | 766 | 3.572175180786021e-01 |
| 127 | 0.000000000000000e+00 | 447 | 1.008125875904472e+00 | 767 | 3.460422935204829e-01 |
| 128 | 0.000000000000000e+00 | 448 | 1.007972441990187e+00 | 768 | 3.349719592361666e-01 |
| 129 | 0.000000000000000e+00 | 449 | 1.007811036585595e+00 | 769 | 3.240095704645023e-01 |
| 130 | 0.000000000000000e+00 | 450 | 1.007642728426847e+00 | 770 | 3.131603499997996e-01 |
| 131 | 0.000000000000000e+00 | 451 | 1.007466616298057e+00 | 771 | 3.024270279840051e-01 |
| 132 | 0.000000000000000e+00 | 452 | 1.007283799246233e+00 | 772 | 2.918144694729168e-01 |
| 133 | 0.000000000000000e+00 | 453 | 1.007093045696527e+00 | 773 | 2.813259021832532e-01 |
| 134 | 0.000000000000000e+00 | 454 | 1.006895570408563e+00 | 774 | 2.709659073501196e-01 |
| 135 | 0.000000000000000e+00 | 455 | 1.006690541428116e+00 | 775 | 2.607365124918583e-01 |
| 136 | 0.000000000000000e+00 | 456 | 1.006479018139540e+00 | 776 | 2.506420640291662e-01 |
| 137 | 0.000000000000000e+00 | 457 | 1.006259855360867e+00 | 777 | 2.406837992341654e-01 |
| 138 | 0.000000000000000e+00 | 458 | 1.006034554002353e+00 | 778 | 2.308655379767671e-01 |
| 139 | 0.000000000000000e+00 | 459 | 1.005802269635282e+00 | 779 | 2.211888523410642e-01 |
| 140 | 0.000000000000000e+00 | 460 | 1.005563968008037e+00 | 780 | 2.116567430906184e-01 |
| 141 | 0.000000000000000e+00 | 461 | 1.005318795748286e+00 | 781 | 2.022699854906251e-01 |
| 142 | 0.000000000000000e+00 | 462 | 1.005068272394360e+00 | 782 | 1.930318183054904e-01 |
| 143 | 0.000000000000000e+00 | 463 | 1.004811375053364e+00 | 783 | 1.839431938620024e-01 |
| 144 | 0.000000000000000e+00 | 464 | 1.004549339226336e+00 | 784 | 1.750067399059861e-01 |
| 145 | 0.000000000000000e+00 | 465 | 1.004281457582935e+00 | 785 | 1.662224799248571e-01 |
| 146 | 0.000000000000000e+00 | 466 | 1.004009147462043e+00 | 786 | 1.575921408388593e-01 |
| 147 | 0.000000000000000e+00 | 467 | 1.003731570287893e+00 | 787 | 1.491152519299797e-01 |
| 148 | 0.000000000000000e+00 | 468 | 1.003450063374329e+00 | 788 | 1.407938190608664e-01 |
| 149 | 0.000000000000000e+00 | 469 | 1.003163845364970e+00 | 789 | 1.326272948938113e-01 |
| 150 | 0.000000000000000e+00 | 470 | 1.002874411368430e+00 | 790 | 1.246171387327525e-01 |
| 151 | 0.000000000000000e+00 | 471 | 1.002580975161843e+00 | 791 | 1.167626546016197e-01 |
| 152 | 0.000000000000000e+00 | 472 | 1.002284871786226e+00 | 792 | 1.090651518336202e-01 |
| 153 | 0.000000000000000e+00 | 473 | 1.001985425397567e+00 | 793 | 1.015233140203748e-01 |
| 154 | 0.000000000000000e+00 | 474 | 1.001684244734497e+00 | 794 | 9.413813765275064e-02 |
| 155 | 0.000000000000000e+00 | 475 | 1.001380551217109e+00 | 795 | 8.690807412770696e-02 |
| 156 | 0.000000000000000e+00 | 476 | 1.001075613053728e+00 | 796 | 7.983354189816511e-02 |
| 157 | 0.000000000000000e+00 | 477 | 1.000768859280338e+00 | 797 | 7.291293184312803e-02 |
| 158 | 0.000000000000000e+00 | 478 | 1.000461955079660e+00 | 798 | 6.614647812869151e-02 |
| 159 | 0.000000000000000e+00 | 479 | 1.000153907612080e+00 | 799 | 5.953239090915557e-02 |
| 160 | 0.000000000000000e+00 | 480 | 9.998461160718275e-01 | 800 | 5.307133911821893e-02 |
| 161 | 0.000000000000000e+00 | 481 | 9.995382582242990e-01 | 801 | 4.676102915752826e-02 |
| 162 | 0.000000000000000e+00 | 482 | 9.992317314100975e-01 | 802 | 4.060106462625085e-02 |
| 163 | 0.000000000000000e+00 | 483 | 9.989255426466267e-01 | 803 | 3.458896350634671e-02 |
| 164 | 0.000000000000000e+00 | 484 | 9.986213520769593e-01 | 804 | 2.872481423270595e-02 |
| 165 | 0.000000000000000e+00 | 485 | 9.983185871761977e-01 | 805 | 2.300642061077823e-02 |
| 166 | 0.000000000000000e+00 | 486 | 9.980185087055744e-01 | 806 | 1.743398319747869e-02 |
| 167 | 0.000000000000000e+00 | 487 | 9.977203369515556e-01 | 807 | 1.200477020244778e-02 |
| 168 | 0.000000000000000e+00 | 488 | 9.974256691222113e-01 | 808 | 6.717697635290676e-03 |
| 169 | 0.000000000000000e+00 | 489 | 9.971338271912752e-01 | 809 | 1.570315476576933e-03 |
| 170 | 0.000000000000000e+00 | 490 | 9.968461329825753e-01 | 810 | -3.438268661133170e-03 |
| 171 | 0.000000000000000e+00 | 491 | 9.965617986382630e-01 | 811 | -8.310425696208936e-03 |
| 172 | 0.000000000000000e+00 | 492 | 9.962823025614079e-01 | 812 | -1.304581363895818e-02 |
| 173 | 0.000000000000000e+00 | 493 | 9.960068616185641e-01 | 813 | -1.764740541599136e-02 |
| 174 | 0.000000000000000e+00 | 494 | 9.957367951478069e-01 | 814 | -2.211581608120528e-02 |
| 175 | 0.000000000000000e+00 | 495 | 9.954712635321227e-01 | 815 | -2.645425077642105e-02 |
| 176 | 0.000000000000000e+00 | 496 | 9.952116634297566e-01 | 816 | -3.066341518682682e-02 |
| 177 | 0.000000000000000e+00 | 497 | 9.949572854565533e-01 | 817 | -3.474585776145929e-02 |
| 178 | 0.000000000000000e+00 | 498 | 9.947093441694513e-01 | 818 | -3.870195772538443e-02 |
| 179 | 0.000000000000000e+00 | 499 | 9.944668184371617e-01 | 819 | -4.253455686336916e-02 |
| 180 | 3.504438130619497e-03 | 500 | 9.942312024833810e-01 | 820 | -4.624456893420224e-02 |
| 181 | 6.758124889697787e-03 | 501 | 9.940016434044485e-01 | 821 | -4.983427241976235e-02 |
| 182 | 8.943474189364638e-03 | 502 | 9.937790866568735e-01 | 822 | -5.330406164482222e-02 |
| 183 | 1.108558877807282e-02 | 503 | 9.935626893131695e-01 | 823 | -5.665655641133161e-02 |
| 184 | 1.336381425803020e-02 | 504 | 9.933539244159140e-01 | 824 | -5.989191912667712e-02 |
| 185 | 1.592127815153420e-02 | 505 | 9.931516528513824e-01 | 825 | -6.301349420724424e-02 |
| 186 | 1.872896194002638e-02 | 506 | 9.929569112537944e-01 | 826 | -6.602189797715241e-02 |
| 187 | 2.183057564646272e-02 | 507 | 9.927688708468421e-01 | 827 | -6.891994783815969e-02 |
| 188 | 2.520980565928884e-02 | 508 | 9.925887208794144e-01 | 828 | -7.170797002873608e-02 |
| 189 | 2.889548499582958e-02 | 509 | 9.924152398352751e-01 | 829 | -7.438785600211463e-02 |
| 190 | 3.288072750732215e-02 | 510 | 9.922495028313456e-01 | 830 | -7.695980775819990e-02 |
| 191 | 3.718640251368464e-02 | 511 | 9.920906151219310e-01 | 831 | -7.942625026703617e-02 |
| 192 | 4.180950541438362e-02 | 512 | 9.919396217291009e-01 | 832 | -8.178792716809512e-02 |
| 193 | 4.676538412257621e-02 | 513 | 9.917952720685562e-01 | 833 | -8.404742152815330e-02 |
| 194 | 5.205244216987966e-02 | 514 | 9.916586940166509e-01 | 834 | -8.620493821803937e-02 |
| 195 | 5.768176015814392e-02 | 515 | 9.915288011543961e-01 | 835 | -8.826309993000785e-02 |
| 196 | 6.365188111381365e-02 | 516 | 9.914064705361564e-01 | 836 | -9.022171021406450e-02 |
| 197 | 6.997033405748826e-02 | 517 | 9.912905118607647e-01 | 837 | -9.208300062891550e-02 |
| 198 | 7.663508305536153e-02 | 518 | 9.911819240108124e-01 | 838 | -9.384684920715425e-02 |
| 199 | 8.365057236623055e-02 | 519 | 9.910795247770178e-01 | 839 | -9.551545820689084e-02 |
| 200 | 9.101379000888035e-02 | 520 | 9.909842592301273e-01 | 840 | -9.708911135857967e-02 |
| 201 | 9.872637505002878e-02 | 521 | 9.908947858311721e-01 | 841 | -9.857014566194627e-02 |
| 202 | 1.067840430193724e-01 | 522 | 9.908120376822228e-01 | 842 | -9.995864571932928e-02 |
| 203 | 1.151858295527293e-01 | 523 | 9.907348826312993e-01 | 843 | -1.012568997532046e-01 |
| 204 | 1.239260664828526e-01 | 524 | 9.906640366554630e-01 | 844 | -1.024650162703341e-01 |
| 205 | 1.330015240938615e-01 | 525 | 9.905980602136440e-01 | 845 | -1.035849043557610e-01 |
| 206 | 1.424052801149985e-01 | 526 | 9.905379830873822e-01 | 846 | -1.046162812518618e-01 |
| 207 | 1.521320683467849e-01 | 527 | 9.904825650601110e-01 | 847 | -1.055612509762041e-01 |
| 208 | 1.621735416384834e-01 | 528 | 9.904324484666853e-01 | 848 | -1.064196358069465e-01 |
| 209 | 1.725221947208913e-01 | 529 | 9.903862280081246e-01 | 849 | -1.071937180231978e-01 |
| 210 | 1.831680872008943e-01 | 530 | 9.903448913950654e-01 | 850 | -1.078834293141886e-01 |
| 211 | 1.941021906320988e-01 | 531 | 9.903071270768942e-01 | 851 | -1.084908852561722e-01 |
| 212 | 2.053139897833021e-01 | 532 | 9.902734448815004e-01 | 852 | -1.090163960055733e-01 |
| 213 | 2.167933432162879e-01 | 533 | 9.902424418296485e-01 | 853 | -1.094621746650760e-01 |
| 214 | 2.285283969438072e-01 | 534 | 9.902151896501201e-01 | 854 | -1.098276928170364e-01 |
| 215 | 2.405069849650012e-01 | 535 | 9.901902265696609e-01 | 855 | -1.101145827722143e-01 |
| 216 | 2.527161881181577e-01 | 536 | 9.901680598867444e-01 | 856 | -1.103225838568563e-01 |
| 217 | 2.651434678028531e-01 | 537 | 9.901474648912307e-01 | 857 | -1.104537426996073e-01 |
| 218 | 2.777758503744847e-01 | 538 | 9.901293790312005e-01 | 858 | -1.105086416532167e-01 |
| 219 | 2.905996158436682e-01 | 539 | 9.901122448734595e-01 | 859 | -1.104898474883336e-01 |
| 220 | 3.036004651973109e-01 | 540 | 9.900966926051257e-01 | 860 | -1.103972812085189e-01 |
| 221 | 3.167640325043893e-01 | 541 | 9.900814722948890e-01 | 861 | -1.102332261219973e-01 |
| 222 | 3.300763764703638e-01 | 542 | 9.900674746047259e-01 | 862 | -1.099969655786929e-01 |
| 223 | 3.435228672445627e-01 | 543 | 9.900532105829365e-01 | 863 | -1.096903559498340e-01 |
| 224 | 3.570886786795506e-01 | 544 | 9.900394023736973e-01 | 864 | -1.093135588677235e-01 |
| 225 | 3.707583717376236e-01 | 545 | 9.900248320990181e-01 | 865 | -1.088690135027248e-01 |
| 226 | 3.845172335531009e-01 | 546 | 9.900103500807507e-01 | 866 | -1.083570625865931e-01 |
| 227 | 3.983499612526493e-01 | 547 | 9.899945557067980e-01 | 867 | -1.077799961121537e-01 |
| 228 | 4.122417888542512e-01 | 548 | 9.899782261038060e-01 | 868 | -1.071382314127799e-01 |
| 229 | 4.261772971493010e-01 | 549 | 9.899600605054418e-01 | 869 | -1.064342821660323e-01 |
| 230 | 4.401421815729068e-01 | 550 | 9.899410789223559e-01 | 870 | -1.056686838192766e-01 |
| 231 | 4.541216995958746e-01 | 551 | 9.899200050208486e-01 | 871 | -1.048438825017798e-01 |
| 232 | 4.681017110048007e-01 | 552 | 9.898975138787787e-01 | 872 | -1.039596356759290e-01 |
| 233 | 4.820662943337458e-01 | 553 | 9.898725363809847e-01 | 873 | -1.030183804850491e-01 |
| 234 | 4.960008074926423e-01 | 554 | 9.898462068764986e-01 | 874 | -1.020202684361974e-01 |
| 235 | 5.098915423728255e-01 | 555 | 9.898170267411330e-01 | 875 | -1.009680221406219e-01 |
| 236 | 5.237263500445715e-01 | 556 | 9.897859195209235e-01 | 876 | -9.986271288271026e-02 |
| 237 | 5.374905090437071e-01 | 557 | 9.897520286480039e-01 | 877 | -9.870736514214426e-02 |
| 238 | 5.511703155400274e-01 | 558 | 9.897162195263046e-01 | 878 | -9.750284620437569e-02 |
| 239 | 5.647535885752191e-01 | 559 | 9.896772011542693e-01 | 879 | -9.625155313139856e-02 |
| 240 | 5.782289366005894e-01 | 560 | 9.896362652966239e-01 | 880 | -9.495377758870335e-02 |
| 241 | 5.915799587176216e-01 | 561 | 9.895923617530382e-01 | 881 | -9.361232867223340e-02 |
| 242 | 6.047954625702541e-01 | 562 | 9.895463342815009e-01 | 882 | -9.222795761428432e-02 |
| 243 | 6.178641647786525e-01 | 563 | 9.894972124952000e-01 | 883 | -9.080411291245728e-02 |
| 244 | 6.307755329382612e-01 | 564 | 9.894462830852175e-01 | 884 | -8.934164364321417e-02 |
| 245 | 6.435179268559957e-01 | 565 | 9.893923680007577e-01 | 885 | -8.784374452959058e-02 |
| 246 | 6.560830137986515e-01 | 566 | 9.893365186903538e-01 | 886 | -8.631137544905677e-02 |
| 247 | 6.684616478236647e-01 | 567 | 9.892779555818088e-01 | 887 | -8.474728946770714e-02 |
| 248 | 6.806438831866077e-01 | 568 | 9.892178658748239e-01 | 888 | -8.315237353264004e-02 |
| 249 | 6.926196927377140e-01 | 569 | 9.891551701556196e-01 | 889 | -8.152964005319532e-02 |
| 250 | 7.043814340356083e-01 | 570 | 9.890911247701029e-01 | 890 | -7.988010693411644e-02 |
| 251 | 7.159201919962611e-01 | 571 | 9.890247977602895e-01 | 891 | -7.820711420768856e-02 |
| 252 | 7.272298755824693e-01 | 572 | 9.889574384705896e-01 | 892 | -7.651178225890490e-02 |
| 253 | 7.383028603058783e-01 | 573 | 9.888882480852147e-01 | 893 | -7.479758913001458e-02 |
| 254 | 7.491330971820804e-01 | 574 | 9.888182771263505e-01 | 894 | -7.306581273272733e-02 |
| 255 | 7.597145736971065e-01 | 575 | 9.887466261142505e-01 | 895 | -7.131966266443390e-02 |
| 256 | 7.700431275216928e-01 | 576 | 9.886749404176028e-01 | 896 | -6.955996868581379e-02 |
| 257 | 7.801137731566502e-01 | 577 | 9.886022219397892e-01 | 897 | -6.778962269634495e-02 |
| 258 | 7.899235162194718e-01 | 578 | 9.885294200392185e-01 | 898 | -6.600961519440657e-02 |
| 259 | 7.994681492797084e-01 | 579 | 9.884560159878955e-01 | 899 | -6.422303545004304e-02 |
| 260 | 8.087449817124315e-01 | 580 | 9.883835200189653e-01 | 900 | -6.243104027829089e-02 |
| 261 | 8.177505366573690e-01 | 581 | 9.883107693992456e-01 | 901 | -6.063717235243996e-02 |
| 262 | 8.264839911291133e-01 | 582 | 9.882390300097154e-01 | 902 | -5.884271749745559e-02 |
| 263 | 8.349435141989714e-01 | 583 | 9.881678007807095e-01 | 903 | -5.705123152423822e-02 |
| 264 | 8.431297226886574e-01 | 584 | 9.880984675859855e-01 | 904 | -5.526334794593565e-02 |
| 265 | 8.510420440685049e-01 | 585 | 9.880300303653743e-01 | 905 | -5.348162036097223e-02 |
| 266 | 8.586812520174252e-01 | 586 | 9.879637979933339e-01 | 906 | -5.170690802826666e-02 |
| 267 | 8.660476534563131e-01 | 587 | 9.878991990245439e-01 | 907 | -4.994225863256500e-02 |
| 268 | 8.731437835983047e-01 | 588 | 9.878376489591358e-01 | 908 | -4.818891490266687e-02 |
| 269 | 8.799712722567934e-01 | 589 | 9.877781920433015e-01 | 909 | -4.645022292934329e-02 |
| 270 | 8.865337190368053e-01 | 590 | 9.877221556193183e-01 | 910 | -4.472698045914417e-02 |
| 271 | 8.928338316495705e-01 | 591 | 9.876689801932402e-01 | 911 | -4.302203371734278e-02 |
| 272 | 8.988755194372424e-01 | 592 | 9.876201238703241e-01 | 912 | -4.133575417353826e-02 |
| 273 | 9.046620597741508e-01 | 593 | 9.875746535410983e-01 | 913 | -3.967067992672979e-02 |
| 274 | 9.101986790930605e-01 | 594 | 9.875338926695315e-01 | 914 | -3.802742318209150e-02 |
| 275 | 9.154896280575360e-01 | 595 | 9.874973205882319e-01 | 915 | -3.640892406933019e-02 |
| 276 | 9.205404627076625e-01 | 596 | 9.874663281231737e-01 | 916 | -3.481597793538342e-02 |
| 277 | 9.253567968750328e-01 | 597 | 9.874399717110517e-01 | 917 | -3.325098858986303e-02 |
| 278 | 9.299449872197452e-01 | 598 | 9.874197049003676e-01 | 918 | -3.171440372994384e-02 |
| 279 | 9.343112966094085e-01 | 599 | 9.874049060317581e-01 | 919 | -3.020862738245264e-02 |
| 280 | 9.384634163826711e-01 | 600 | 9.873969162747074e-01 | 920 | -2.873390902713615e-02 |
| 281 | 9.424071613625743e-01 | 601 | 9.873949921033299e-01 | 921 | -2.729263737090799e-02 |
| 282 | 9.461498120176854e-01 | 602 | 9.874004750404033e-01 | 922 | -2.588471937157619e-02 |
| 283 | 9.496984081652284e-01 | 603 | 9.874126414437681e-01 | 923 | -2.451227449041489e-02 |
| 284 | 9.530600909726344e-01 | 604 | 9.874329809802893e-01 | 924 | -2.317526315266411e-02 |
| 285 | 9.562427177295731e-01 | 605 | 9.874606249127551e-01 | 925 | -2.187570925786862e-02 |
| 286 | 9.592539757044516e-01 | 606 | 9.874969061399389e-01 | 926 | -2.061355417582714e-02 |
| 287 | 9.621011497566145e-01 | 607 | 9.875412739766566e-01 | 927 | -1.939065975232718e-02 |
| 288 | 9.647915124057732e-01 | 608 | 9.875949843246187e-01 | 928 | -1.820666399965468e-02 |
| 289 | 9.673331975559332e-01 | 609 | 9.876571983429736e-01 | 929 | -1.706288556735433e-02 |
| 290 | 9.697327413658168e-01 | 610 | 9.877295459610343e-01 | 930 | -1.595856621933800e-02 |
| 291 | 9.719990112065752e-01 | 611 | 9.878111744348976e-01 | 931 | -1.489528159506296e-02 |
| 292 | 9.741378617337759e-01 | 612 | 9.879032023766432e-01 | 932 | -1.387263484323160e-02 |
| 293 | 9.761574317374303e-01 | 613 | 9.880051626345804e-01 | 933 | -1.289205910303846e-02 |
| 294 | 9.780617486993630e-01 | 614 | 9.881185638915363e-01 | 934 | -1.195270300743193e-02 |
| 295 | 9.798613526271561e-01 | 615 | 9.882423270582524e-01 | 935 | -1.105520547739066e-02 |
| 296 | 9.815583336011345e-01 | 616 | 9.883778520531268e-01 | 936 | -1.019827182163307e-02 |
| 297 | 9.831750948772566e-01 | 617 | 9.885247606051014e-01 | 937 | -9.382480860899108e-03 |
| 298 | 9.847362453480265e-01 | 618 | 9.886843467692753e-01 | 938 | -8.606586039759667e-03 |
| 299 | 9.863553220272523e-01 | 619 | 9.888556356037892e-01 | 939 | -7.871422820642307e-03 |
| 300 | 9.879313024174022e-01 | 620 | 9.890401927796704e-01 | 940 | -7.175885277771099e-03 |
| 301 | 9.890757868707590e-01 | 621 | 9.892374835404283e-01 | 941 | -6.520419726599805e-03 |
| 302 | 9.901827419790691e-01 | 622 | 9.894488374513719e-01 | 942 | -5.903403814095400e-03 |
| 303 | 9.912523092989319e-01 | 623 | 9.896735637374597e-01 | 943 | -5.324608721716763e-03 |
| 304 | 9.922861082472264e-01 | 624 | 9.899131011580791e-01 | 944 | -4.782469904501813e-03 |
| 305 | 9.932837131068657e-01 | 625 | 9.901668953434221e-01 | 945 | -4.276873767639064e-03 |
| 306 | 9.942466438407230e-01 | 626 | 9.904366799536263e-01 | 946 | -3.806285163352241e-03 |
| 307 | 9.951746430061121e-01 | 627 | 9.907216425865902e-01 | 947 | -3.370454877988472e-03 |
| 308 | 9.960694269553644e-01 | 628 | 9.910230654424902e-01 | 948 | -2.967607624839524e-03 |
| 309 | 9.969306036935497e-01 | 629 | 9.913408767719142e-01 | 949 | -2.597106916737789e-03 |
| 310 | 9.977600196406868e-01 | 630 | 9.916767775088281e-01 | 950 | -2.257081732588478e-03 |
| 311 | 9.985573503390627e-01 | 631 | 9.920297273323891e-01 | 951 | -1.946591608170231e-03 |
| 312 | 9.993242684851855e-01 | 632 | 9.924015177880798e-01 | 952 | -1.664473096973725e-03 |
| 313 | 1.000060686758758e+00 | 633 | 9.927921871265732e-01 | 953 | -1.409209039594871e-03 |
| 314 | 1.000768505317086e+00 | 634 | 9.932031606606759e-01 | 954 | -1.180256894474562e-03 |
| 315 | 1.001447278873483e+00 | 635 | 9.936337788972491e-01 | 955 | -9.747166718929050e-04 |
| 316 | 1.002098854400575e+00 | 636 | 9.940860378486615e-01 | 956 | -7.929180432976445e-04 |
| 317 | 1.002723127308148e+00 | 637 | 9.945597525471822e-01 | 957 | -6.262931535610879e-04 |
| 318 | 1.003321903663793e+00 | 638 | 9.950565724564597e-01 | 958 | -4.619898752628163e-04 |
| 319 | 1.003894772403371e+00 | 639 | 9.955761256313581e-01 | 959 | -2.353032150516754e-04 |