Draft **ETSI EN 300 468** V1.19.0 (2024-11)

**EUROPEAN STANDARD**

**Digital Video Broadcasting (DVB);**
**Specification for Service Information (SI) in DVB systems**

**EBU D/3**

*Important notice*

The present document can be downloaded from the
ETSI Search & Browse Standards application.

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or
print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any
existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI
deliverable is the one made publicly available in PDF format on ETSI deliver repository.

Users should be aware that the present document may be revised or have its status changed,
this information is available in the Milestones listing.

If you find errors in the present document, please send your comments to
the relevant service listed under Committee Support Staff.

If you find a security vulnerability in the present document, please report it through our
Coordinated Vulnerability Disclosure (CVD) program.

*Notice of disclaimer & limitation of liability*

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of
experience to understand and interpret its content in accordance with generally accepted engineering or
other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not
limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property
rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages
for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use
of or inability to use the software.

*Copyright Notification*

# Contents

# Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the [ETSI IPR online database](#).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

**DECT™**, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

# Foreword

This draft European Standard (EN) has been produced by Joint Technical Committee (JTC) Broadcast of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI), and is now submitted for the combined Public Enquiry and Vote phase of the ETSI EN Approval Procedure.

NOTE:     The EBU/ETSI JTC Broadcast was established in 1990 to co-ordinate the drafting of standards in the specific field of broadcasting and related fields. Since 1995 the JTC Broadcast became a tripartite body by including in the Memorandum of Understanding also CENELEC, which is responsible for the standardization of radio and television receivers. The EBU is a professional association of broadcasting organizations whose work includes the co-ordination of its members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has active members in about 60 countries in the European broadcasting area; its headquarters is in Geneva.

European Broadcasting Union
CH-1218 GRAND SACONNEX (Geneva)
Switzerland
Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

The DVB Project is an industry-led consortium of broadcasters, manufacturers, network operators, software developers, regulators and others from around the world committed to designing open, interoperable technical specifications for the global delivery of digital media and broadcast services. DVB specifications cover all aspects of digital television from transmission through interfacing, conditional access and interactivity for digital video, audio and data. The consortium came together in 1993.

| Proposed national transposition dates | |
|---|---|
| Date of latest announcement of this EN (doa): | 3 months after ETSI publication |
| Date of latest publication of new National Standard or endorsement of this EN (dop/e): | 6 months after doa |
| Date of withdrawal of any conflicting National Standard (dow): | 6 months after doa |

# Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

# 1        Scope

The present document specifies the Service Information (SI) data which forms a part of Digital Video Broadcasting (DVB) bitstreams, in order that the user can be provided with information to assist in selection of services and/or events within the bitstream, and so that the Integrated Receiver Decoder (IRD) can automatically configure itself for the selected service. SI data for automatic configuration is mostly specified within ISO/IEC 13818-1 [1] as Program Specific Information (PSI).

The present document specifies additional data which complements the PSI by providing data to aid automatic tuning of IRDs, and additional information intended for display to the user. The manner of presentation of the information is not specified in the present document, and IRD manufacturers have freedom to choose appropriate presentation methods.

It is expected that Electronic Programme Guide (EPG) will be a feature of Digital TeleVision (TV) transmissions.

The definition of an EPG is outside the scope of the present document (i.e. the SI specification), but the data contained within the SI specified in the present document may be used as the basis for an EPG.

Rules of operation for the implementation of the present document are specified in ETSI TS 101 211 [i.1].

# 2        References

## 2.1      Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference/.

NOTE:      While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1]        ISO/IEC 13818-1: "Information technology - Generic coding of moving pictures and associated audio information - Part 1: Systems".

[2]        ETSI EN 300 743: "Digital Video Broadcasting (DVB); Subtitling systems".

[3]        ETSI EN 301 192: "Digital Video Broadcasting (DVB); DVB specification for data broadcasting".

[4]        ETSI EN 301 210: "Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for Digital Satellite News Gathering (DSNG) and other contribution applications by satellite".

[5]        ETSI EN 301 775: "Digital Video Broadcasting (DVB); Specification for the carriage of Vertical Blanking Information (VBI) data in DVB bitstreams".

[6]        ETSI EN 301 790: "Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems".

[7]        ETSI EN 302 307-1: "Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 1: DVB-S2".

[8]        ETSI EN 302 307-2: "Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 2: DVB-S2 Extensions (DVB-S2X)".

[9]     ETSI EN 302 769: "Digital Video Broadcasting (DVB); Frame structure channel coding and modulation for a second generation digital transmission system for cable systems (DVB-C2)".

[10]    ETSI EN 302 583: "Digital Video Broadcasting (DVB); Framing Structure, channel coding and modulation for Satellite Services to Handheld devices (SH) below 3 GHz".

[11]    ETSI EN 302 755: "Digital Video Broadcasting (DVB); Frame structure channel coding and modulation for a second generation digital terrestrial television broadcasting system (DVB-T2)".

[12]    ETSI EN 303 560: "Digital Video Broadcasting (DVB); TTML subtitling systems".

[13]    ETSI ES 201 812: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.0.3".

[14]    ETSI TS 101 154: "Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in Broadcast and Broadband Applications".

[15]    ETSI TS 101 162: "Digital Video Broadcasting (DVB); Allocation of identifiers and codes for Digital Video Broadcasting (DVB) systems".

[16]    ETSI TS 101 547-2: "Digital Video Broadcasting (DVB); Plano-stereoscopic 3DTV; Part 2: Frame Compatible Plano-stereoscopic 3DTV".

[17]    ETSI TS 101 547-3: "Digital Video Broadcasting (DVB); Plano-stereoscopic 3DTV; Part 3: HDTV Service Compatible Plano-stereoscopic 3DTV".

[18]    ETSI TS 101 547-4: "Digital Video Broadcasting (DVB); Plano-stereoscopic 3DTV; Part 4: Service frame compatible Plano-stereoscopic 3DTV for HEVC coded services".

[19]    ETSI TS 102 005: "Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in DVB services delivered directly over IP protocols".

[20]    ETSI TS 102 006: "Digital Video Broadcasting (DVB); Specification for System Software Update in DVB Systems".

[21]    ETSI TS 102 323: "Digital Video Broadcasting (DVB); Carriage and signalling of TV-Anytime information in DVB transport streams".

[22]    ETSI TS 102 770: "Digital Video Broadcasting (DVB); System Renewability Messages (SRM) in DVB Systems".

[23]    ETSI TS 102 772: "Digital Video Broadcasting (DVB); Specification of Multi-Protocol Encapsulation - inter-burst Forward Error Correction (MPE-iFEC)".

[24]    ETSI TS 102 773: "Digital Video Broadcasting (DVB); Modulator Interface (T2-MI) for a second generation digital terrestrial television broadcasting system (DVB-T2)".

[25]    ETSI TS 102 809: "Digital Video Broadcasting (DVB); Signalling and carriage of interactive applications and services in Hybrid Broadcast/Broadband environments".

[26]    ETSI TS 102 812: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.1.3".

[27]    ETSI TS 102 825 (parts 1 to 5, 7, 9 and 10): "Digital Video Broadcasting (DVB); Content Protection and Copy Management (DVB-CPCM)".

[28]    ETSI EN 300 231: "Television systems; Specification of the domestic video Programme Delivery Control system (PDC)".

[29]    ETSI EN 300 401: "Radio Broadcasting Systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers".

[30]    ETSI EN 300 706: "Enhanced Teletext specification".

[31]    EN 50221: "Common interface specification for conditional access and other digital video broadcasting decoder applications", (produced by CENELEC).

[32] IETF RFC 2045: "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies".

[33] IETF RFC 3986: "Uniform Resource Identifiers (URI): Generic Syntax".

[34] ISO 3166 (all parts): "Codes for the representation of names of countries and their subdivisions".

[35] ISO 639-2: "Codes for the representation of names of languages - Part 2: Alpha-3 code".

[36] ISO 8601-1: "Date and time -- Representations for information interchange -- Part 1: Basic rules".

[37] ISO/IEC 6937: "Information technology - Coded graphic character set for text communication - Latin alphabet".

[38] ISO/IEC 8859-1: "Information technology - 8-bit single-byte coded graphic character sets - Part 1: Latin alphabet No. 1".

[39] ISO/IEC 8859-2: "Information technology - 8-bit single-byte coded graphic character sets - Part 2: Latin alphabet No. 2".

[40] ISO/IEC 8859-3: "Information technology - 8-bit single-byte coded graphic character sets - Part 3: Latin alphabet No. 3".

[41] ISO/IEC 8859-4: "Information technology - 8-bit single-byte coded graphic character sets - Part 4: Latin alphabet No. 4".

[42] ISO/IEC 8859-5: "Information technology - 8-bit single-byte coded graphic character sets - Part 5: Latin/Cyrillic alphabet".

[43] ISO/IEC 8859-6: "Information technology - 8-bit single-byte coded graphic character sets - Part 6: Latin/Arabic alphabet".

[44] ISO/IEC 8859-7: "Information technology - 8-bit single-byte coded graphic character sets - Part 7: Latin/Greek alphabet".

[45] ISO/IEC 8859-8: "Information technology - 8-bit single-byte coded graphic character sets - Part 8: Latin/Hebrew alphabet".

[46] ISO/IEC 8859-9: "Information technology - 8-bit single-byte coded graphic character sets - Part 9: Latin alphabet No. 5".

[47] ISO/IEC 8859-10: "Information technology - 8-bit single-byte coded graphic character sets - Part 10: Latin alphabet No. 6".

[48] ISO/IEC 8859-11: "Information technology - 8-bit single-byte coded graphic character sets - Part 11: Latin/Thai alphabet".

[49] ISO/IEC 8859-13: "Information technology - 8-bit single-byte coded graphic character sets - Part 13: Latin alphabet No. 7".

[50] ISO/IEC 8859-14: "Information technology - 8-bit single-byte coded graphic character sets - Part 14: Latin alphabet No. 8 (Celtic)".

[51] ISO/IEC 8859-15: "Information technology - 8-bit single-byte coded graphic character sets - Part 15: Latin alphabet No. 9".

[52] ISO/IEC 10646: "Information technology - Universal Coded Character Set (UCS)".

[53] GB-2312-1980: "Code of Chinese graphic character set for information interchange -- primary set".

[54] KS X 1001-2014: "Code for Information Interchange (Hangeul and Hanja)", Korean Agency for Technology and Standards, 2014.

NOTE: See https://standard.go.kr/KSCI/standardIntro/getStandardSearchView.do?ksNo=KSX1001. This document has been published in Korean only.

[55]        IEEE 754™-2019: "Standard for Floating-Point, 2019.

[56]        ETSI EN 301 545-2: "Digital Video Broadcasting (DVB); Second Generation DVB Interactive Satellite System (DVB-RCS2); Part 2: Lower Layers for Satellite standard".

[57]        DVB BlueBook A180: "Native IP Broadcasting".

## 2.2        Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE 1:   While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1]        ETSI TS 101 211: "Digital Video Broadcasting (DVB); Implementation and usage of Service Information (SI)".

[i.2]        ETSI TS 102 727: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2.2".

[i.3]        ETSI TS 103 205: "Digital Video Broadcasting (DVB); Extensions to the CI Plus™ Specification".

[i.4]        ETSI TS 103 286-2: "Digital Video Broadcasting (DVB); Companion Screens and Streams; Part 2: Content Identification and Media Synchronization".

[i.5]        ETSI TR 102 825 (parts 6, 8, 11 to 13): "Digital Video Broadcasting (DVB); Content Protection and Copy Management (DVB-CPCM)".

[i.6]        ATIS 0800006: "IIF Default Scrambling Algorithm (IDSA) IPTV Interoperability Specification".

[i.7]        IEC 61883 (parts 1 and 4): "Consumer audio/video equipment - Digital interface".

[i.8]        IEEE 1394.1™: "IEEE Standard for High Performance Serial Bus Bridges".

[i.9]        Directive (EU) 2018/1808 of the European Parliament and of the Council of 14 November 2018 amending Directive 2010/13/EU on the coordination of certain provisions laid down by law, regulation or administrative action in Member States concerning the provision of audiovisual media services (Audiovisual Media Services Directive) in view of changing market realities".

[i.10]       American Institute of Aeronautics and Astronautics, AIAA 2006-6753-Rev1: "Revisiting Space track Report #3: Rev 1".

[i.11]       NGA.STND.0036-1.0.0-WGS84 Department of Defense World Geodetic System 1984: "Its Definition and Relationships with Local Geodetic Systems"; Version 1.0.0, 8 July 2014.

# 3 Definition of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the following terms apply:

**AC-3:** coding of audio using the Dolby AC-3 audio compression method as defined in ETSI TS 101 154 [14], clause 6.2

> NOTE: The service information requirements for AC-3 streams carried in DVB systems are described in annex D. The carriage of AC-3 elementary streams as private data within MPEG systems is described in ETSI TS 101 154 [14], clause 6.2.

**AC-4:** coding of audio using the Dolby AC-4 audio compression method as defined in clause 6.6 and clause 6.7 of ETSI TS 101 154 [14]

> NOTE: The Service Information requirements for AC-4 streams carried in DVB systems are described in annex D. The carriage of AC-4 elementary streams as private data within MPEG systems is described in clause 6.6 and clause 6.7 of ETSI TS 101 154 [14].

**audio preselection:** set of audio programme components representing a version of the audio programme that may be selected by a user for simultaneous decoding

> NOTE: An audio preselection is a sub-selection from all available audio programme components of one audio programme. An audio preselection can be considered the NGA equivalent of audio services in predecessor systems, which each utilized complete mixes.

**audio programme:** complete collection of all audio programme components and a set of accompanying audio preselections

> NOTE: Not all audio programme components of an audio programme are necessarily meant to be presented at the same time. An audio programme may contain audio programme components that are always presented, and it may include optional audio programme components.

**audio programme component:** the smallest addressable unit of an audio programme

**auxiliary NGA stream:** NGA stream delivered using NGA multi-stream delivery, and containing additional audio programme components not contained in the main NGA stream

**bouquet:** collection of services marketed as a single entity

**broadcaster:** organization which assembles a sequence of events or data streams to be delivered to the viewer; the delivery can be based upon a schedule

**cell:** geographical area that is covered with DVB signals delivering one or more particular transport or other DVB streams throughout the area by means of one or more transmitters

> NOTE: The cell may in addition contain repeaters. Two neighbouring cells may be intersecting, or fully overlapping. The *cell_id* that is used to uniquely identify a cell is unique within each *original_network_id*. For hand-over purposes it is more convenient if the transport streams associated with the cell cover exactly the same area, or only one transport stream per cell is used.

**component:** one or more entities which together make up an event

> EXAMPLE: Video, audio, teletext.

**conditional access system:** system to control subscriber access to services, data streams and events

> EXAMPLE: Videoguard, Eurocrypt.

**delivery system:** physical medium by which one or more DVB transport streams are transmitted

> EXAMPLE: Satellite system, wide-band coaxial cable, fibre optics, terrestrial channel of one emitting point.

**dependent stream:** stream or component which relies on another stream or component in order to be rendered as intended

> EXAMPLE: A dependent view of a 3D video, or a receiver-mix audio description stream.

**DTS:** coding of audio using the DTS audio compression method as defined in ETSI TS 101 154 [14], clause 6.3

> NOTE: The service information requirements for DTS are found in annex G.

**DTS-HD:** coding of audio using the DTS-HD audio compression method as defined in ETSI TS 101 154 [14], clause 6.3

> NOTE: The service information requirements for DTS-HD are found in annex G. Note that DTS-HD is a superset of DTS.

**DTS-UHD:** coding of audio using the DTS-UHD audio compression method as defined in ETSI TS 101 154 [14], clause 6.9

> NOTE: The service information requirements for DTS-UHD are found in annex G.

**DVB transport stream:** MPEG-2 transport stream (ISO/IEC 13818-1 [1]) containing the mandatory service information signalling as defined in the present document

> NOTE: It is recommended that the service information implementation specification ETSI TS 101 211 [i.1] is additionally followed. It defines further requirements for the signalling to help improve the quality of experience for viewers.

**event:** grouping of elementary broadcast data streams with a defined start time and duration belonging to a common service

> EXAMPLE: First half of a football match, News Flash, first part of an entertainment show.

**HEVC_UHDTV_IRD:** initial ultra high definition IRD profile defined in ETSI TS 101 154 [14]

**main NGA stream:** NGA stream delivered using NGA multi-stream delivery, and containing at least all the audio programme components corresponding to at least one audio preselection

**MPEG-2:** ISO/IEC 13818 [1] set of standards

> NOTE: Systems coding is defined in part 1, video coding is defined in part 2, and audio coding is defined in part 3 of ISO/IEC 13818 [1].

**MPEG-4:** coding of video using the H.264/AVC video compression method as defined in clause 5.6 and clause 5.7 of ETSI TS 101 154 [14], or coding of audio using the AAC, HE-AAC, and HE-AAC v2 audio compression methods as defined in clause 6.4 and clause 6.5 of ETSI TS 101 154 [14]

> NOTE: The service information requirements for MPEG-4 streams carried in DVB systems are described in annex H.

**MPEG-H:** coding of audio using the audio compression method as defined in clause 6.8 of ETSI TS 101 154 [14]

**multi-stream delivery:** method for carrying audio programme components in several NGA streams

> NOTE: E.g. when audio programme components offering additional languages are carried in separate elementary streams to facilitate remultiplexing or service aggregation.

**network:** managed and navigable collection of DVB transport streams transmitted on one or more delivery systems generally based on the same physical medium

> NOTE 1: It is possible to operate both, first and second generation delivery systems in the same network (e.g. DVB-T and DVB-T2).

> NOTE 2: A network is identified by its *network_id*. It might be composed of one or more emitting sites.

**NGA stream:** audio elementary stream containing one or more audio programme components of one audio programme

**original_network_id:** unique identifier of a broadcast platform operator

NOTE: This value is assigned by DVB.

**repeater:** special type of transmitter which receives a terrestrial DVB signal and re-transmits it unchanged

NOTE: Hence it does not support changing of the *cell_id*.

**reserved:** When used in the clause defining the coded bit stream, indicates that the value may be used in the future for ISO defined extensions. Unless otherwise specified within the present document all reserved bits are set to 0b1.

**reserved_future_use:** When used in the clause defining the coded bit stream, indicates that the value may be used in the future for ETSI defined extensions. Unless otherwise specified within the present document all reserved_future_use bits are set to 0b1.

**reserved_zero_future_use:** When used in the clause defining the coded bit stream, indicates that the value may be used in the future for ETSI defined extensions. All reserved_zero_future_use bits are set to 0b0.

**section:** syntactic structure used for mapping all service information defined in ETSI EN 300 468 into ISO/IEC 13818-1 [1] DVB Transport Stream packets

**service:** sequence of programmes under the control of a broadcaster which can be broadcast as part of a schedule

**service_id:** unique identifier of a service within a DVB transport stream

NOTE: In areas where ETSI TS 101 211 [i.1] is applicable in addition to the present document, a *service_id* is also unique within the scope of an *original_network_id*.

**service information:** Describes the delivery system, and the content and scheduling/timing of services and events.

NOTE: It includes MPEG-2 Program Specific Information (PSI) together with DVB-defined extensions.

**single-stream delivery:** method for carrying all audio programme components in a single NGA stream

**sub_table:** collection of sections with the same value of *table_id* and:

- generally: the same *table_id_extension* and *version_number*;

NOTE: The *table_id_extension* field is equivalent to the fourth and fifth byte of a section when the *section_syntax_indicator* is set to a value of 0b1.

- for a NIT: the same *table_id_extension* (*network_id*), and the same *version_number*;

- for a BAT: the same *table_id_extension* (*bouquet_id*), and the same *version_number*;

- for a SDT: the same *table_id_extension* (*transport_stream_id*), the same *original_network_id*, and the same *version_number*;

- for a EIT: the same *table_id_extension* (*service_id*), the same *transport_stream_id*, the same *original_network_id*, and the same *version_number*;

- for a SAT: the same *table_id_extension* (*satellite_table_id and table_count*), and the same *version_number*.

**subcell:** geographical area that is part of the cell's coverage area and that is covered with DVB signals by means of a transposer

NOTE: In conjunction with the *cell_id* the *cell_id_extension* is used to uniquely identify a subcell.

**table:** comprised of a number of sub_tables with the same value of *table_id*

**transmitter:** equipment that emits a modulated DVB stream

**transport stream:** data structure defined in ISO/IEC 13818-1 [1]

NOTE: It is the basis of the DVB standards.

**transport_stream_id:** unique identifier of a DVB transport stream within an original network

**transposer:** special type of repeater which receives a terrestrial DVB signal and re-transmits it on a different frequency

The relationships of some of these definitions are illustrated in the service delivery model in figure 1.
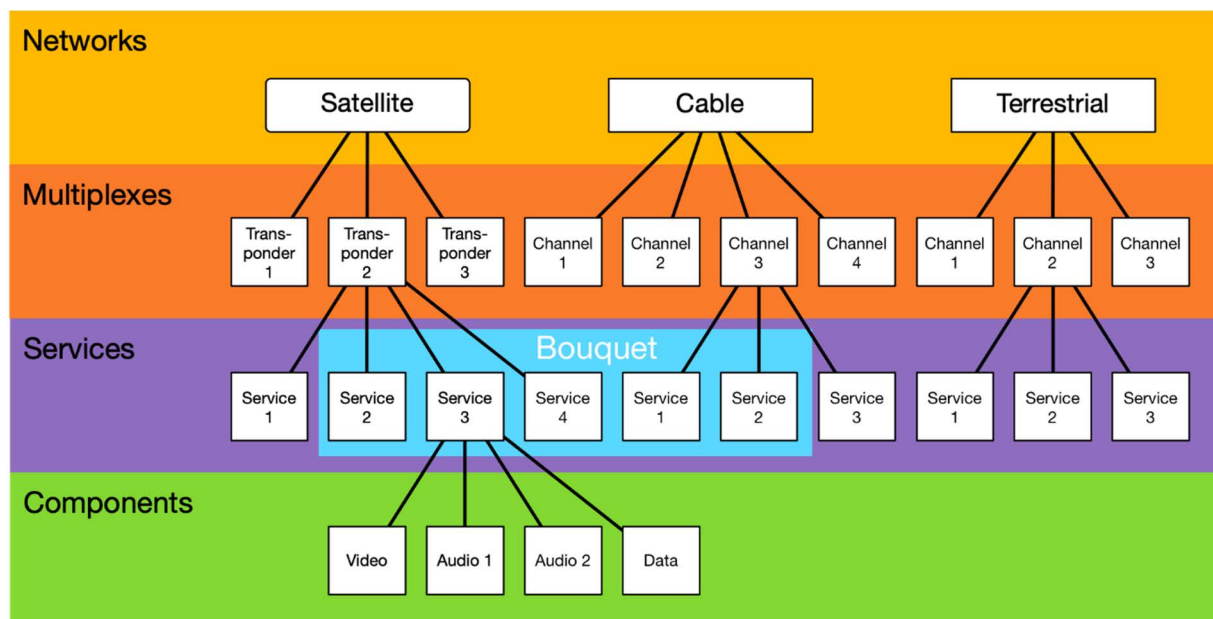


**Figure 1: Digital broadcasting, service delivery model**

## 3.2    Symbols

For the purposes of the present document, the following symbols apply:

0b…          The prefix "0b" denotes a binary number, i.e. an integer number in the polyadic number system with the base 2. Blanks within a binary number string are for ease of reading and have no significance.

0x…          The prefix "0x" denotes a hexadecimal number, i.e. an integer number in the polyadic number system with the base 16. Blanks within a hexadecimal number string are for ease of reading and have no significance.

## 3.3    Abbreviations

For the purposes of the present document, the following abbreviations apply:

| | |
|---|---|
| 128QAM | 128-ary Quadrature Amplitude Modulation |
| 16APSK | 16-ary Amplitude and Phase Shift Keying |
| 16QAM | 16-ary Quadrature Amplitude Modulation |
| 256QAM | 256-ary Quadrature Amplitude Modulation |
| 2D | Two Dimensional |
| 32QAM | 32-ary Quadrature Amplitude Modulation |
| 3DTV | Three-Dimensional Television |
| 64QAM | 64-ary Quadrature Amplitude Modulation |
| 8PSK | 8-ary Phase Shift Keying |
| AAC | Advanced Audio Coding |
| AC-3 | AC-3 audio coding |
| AC-4 | AC-4 audio coding |
| ASCII | American Standard Code for Information Interchange |
| ATIS | Alliance for Telecommunications Industry Solutions |
| AVC | Advanced Video Coding |
| AVS3 | The 3rd generation of the Audio Video coding Standard |
| BAT | Bouquet Association Table |
| BCD | Binary Coded Decimal |

| | |
|---|---|
| BMP | Basic Multilingual Plane |
| bslbf | bit string left bit first |
| CA | Conditional Access |
| CAM | Conditional Access Module |
| CAT | Conditional Access Table |
| CATV | Cable Television |
| CI | Common Interface |
| CID | Content IDentifier |
| CISSA | Common IPTV Software-oriented Scrambling Algorithm |
| CIT | Content Identifier Table |
| CLUT | Colour Look-Up Table |
| CP | Content Protection |
| CPCM | Content Protection Copy Management |
| CR/LF | Carriage Return/Line Feed |
| CRC | Cyclic Redundancy Check |
| CSA1 | Common Scrambling Algorithm Version 1 |
| CSA2 | Common Scrambling Algorithm Version 2 |
| CSA3 | Common Scrambling Algorithm Version 3 |
| DAB | Digital Audio Broadcasting |
| DIT | Discontinuity Information Table |
| DMI | Dynamic Mapping Information |
| DSNG | Digital Satellite News Gathering |
| DTCP | Digital Transmission Content Protection |
| DTS | Decoding Time Stamp |
| DTS | DTS audio coding |
| DTS-HD | DTS-HD audio coding |
| DVB | Digital Video Broadcasting |
| DVB-C | DVB Cable Framing and Modulation |
| DVB-C2 | DVB Cable Framing and Modulation, Second Generation |
| DVB-S | DVB Satellite Framing and Modulation |
| DVB-S2 | DVB Satellite Framing and Modulation, Second Generation |
| DVB-S2X | DVB Satellite Framing and Modulation, Second Generation Extensions |
| DVB-SH | DVB Satellite to Handheld Framing and Modulation |
| DVB-T | DVB Terrestrial Framing and Modulation |
| DVB-T2 | DVB Terrestrial Framing and Modulation, Second Generation |
| DVD | Digital Versatile Disc |
| ECM | Entitlement Control Message |
| EIT | Event Information Table |
| EMM | Entitlement Management Message |
| EPG | Electronic Programme Guide |
| ES | Elementary Stream |
| FC | Frame Compatible |
| FEC | Forward Error Correction |
| FFT | Fast Fourier Transform |
| FLS | Forward Link Signalling |
| FM | Frequency Modulation |
| FTA | Free-To-Air |
| GSE | Generic Stream Encapsulation |
| H.264/AVC | Advanced Video Coding for Generic Audiovisual Services |
| HD | High Definition |
| HDCP | High-bandwidth Digital Content Protection |
| HDR | High Dynamic Range |
| HDTV | High Definition TeleVision |
| HE-AAC | High Efficiency AAC |
| HEVC | High Efficiency Video Coding |
| HFR | High Frame Rate |
| HLG | Hybrid Log Gamma |
| HLG10 | Hybrid Log Gamma with a bit depth of 10 bits |
| HP | High Priority |
| INT | IP/MAC Notification Table |
| IP | Internet Protocol |
| IP/MAC | Internet Protocol/Medium Access Control |

| | |
|---|---|
| IPTV | Internet Protocol Television |
| IRD | Integrated Receiver Decoder |
| ISCR | Input Stream Time Reference |
| ISI | Input Stream Identifier |
| ISSY | Input Stream SYnchronizer |
| JPEG | Joint Photographic Experts Group |
| L1 | Layer One (first or bottom-most layer) |
| LC | Low Complexity |
| LFE | Low Frequency Effects |
| LP | Low Priority |
| lsb | least significant bit |
| MHP | Multimedia Home Platform |
| MI | Modulator Interface |
| MIME | Multipurpose Internet Mail Extensions |
| MISO | Multiple Input, Single Output |
| MJD | Modified Julian Date |
| MPE | Multi-Protocol Encapsulation |
| MPEG | Moving Pictures Expert Group |
| msb | most significant bit |
| MVC | Multi-View video Coding |
| NCR | Network Clock Reference |
| NDA | Non-Disclosure Agreement |
| NGA | Next Generation Audio |
| NIT | Network Information Table |
| NVOD | Near Video On Demand |
| OFDM | Orthogonal Frequency Division Multiplex |
| PAT | Program Association Table |
| PCM | Pulse-Code Modulation |
| PCR | Program Clock Reference |
| PDC | Programme Delivery Control |
| PES | Packetized Elementary Stream |
| PID | Packet IDentifier |
| PIL | Programme Identification Label |
| PLP | Physical Layer Pipe |
| PMT | Program Map Table |
| PNG | Portable Network Graphics |
| PQ10 | Perceptual Quantizer with a bit depth of 10 bits |
| PSI | Program Specific Information |
| PSTN | Public Switched Telephone Network |
| QAM | Quadrature Amplitude Modulation |
| QPSK | Quaternary Phase Shift Keying |
| RAM | Random Access Memory |
| RCS | Return Channel Satellite |
| RCT | Related Content Table |
| RDS | Radio Data System |
| RNT | Resolution Authority Record Notification Table |
| rpchof | remainder polynomial coefficients highest order first |
| RS | Reed-Solomon code |
| RST | Running Status Table |
| SAOC-DE | Spatial Audio Object Coding - Dialogue Enhancement |
| SAT | Satellite Access Table |
| SC | Service Compatible |
| SD | Standard Definition |
| SDR | Standard Dynamic Range |
| SDT | Service Description Table |
| SEI | Supplemental Enhancement Information |
| SFC | Service Frame Compatible |
| SFFI | Superframe Format Indicator |
| SFN | Single Frequency Network |
| SH | Satellite services to Handheld devices |
| SI | Service Information |
| SISO | Single Input, Single Output |

SIT            Selection Information Table
SMATV          Satellite Master Antenna TeleVision
SMI            Storage Media Interoperability
SMPTE          Society of Motion Picture and Television Engineers
SOGI           Service of General Interest
SOSF           Start Of SuperFrame
spfmsbf        single precision floating point value, most significant bit (i.e. the most significant bit of the
               exponent) is first

NOTE:     Single precision floating point is defined in IEEE 754 [55].

SRM            System Renewability Message
SSU            System Software Update
ST             Stuffing Table
tcimsbf        two's complement integer, most significant (sign) bit first
TDM            Time-Domain Multiplex
TDT            Time and Date Table
TFS            Time Frequency Slicing
TLE            Two Line orbital Element
TOC            Table Of Contents
TOT            Time Offset Table
TPS            Transmission Parameter Signalling
TS             DVB Transport Stream
TSDT           Transport Stream Description Table
TTML           Timed Text Markup Language
TV             TeleVision
TVA            TV-Anytime
UCS            Universal Character Set
UECP           Universal Encoder Communication Protocol
UHD            Ultra High Definition
UHDTV          Ultra High Definition TeleVision
uimsbf         unsigned integer most significant bit first
URI            Uniform Resource Identifier
URL            Uniform Resource Locator
USI            Usage State Information
UTC            Universal Time, Co-ordinated
UTF-8          UCS Transformation Format, 8-bit
VBI            Vertical Blanking Interval
VL-SNR         Very Low Signal to Noise Ratio
VPS            Video Programme System
VVC            Versatile Video Coding
WSS            Wide Screen Signalling

# 4        Service information description

ISO/IEC 13818-1 [1] specifies SI which is referred to as PSI. The PSI data provides information to enable automatic configuration of the receiver to demultiplex and decode the various streams of programs within the multiplex.

The PSI data is structured as five types of table. The tables are transmitted in sections:

1)    Program Association Table (PAT):

      for each service in the multiplex, the Program Association Table (PAT) indicates the location (the Packet IDentifier (PID) values of the DVB Transport Stream (TS) packets) of the corresponding Program Map Table (PMT). It also gives the location of the Network Information Table (NIT).

2)    Conditional Access Table (CAT):

      the Conditional Access Table (CAT) provides information on the Conditional Access (CA) systems used in the multiplex; the information is private (not defined within the present document) and dependent on the CA system, but includes the location of the Entitlement Management Message (EMM) stream, when applicable.

3) Program Map Table (PMT):

the PMT identifies and indicates the locations of the streams that make up each service and the location of the Program Clock Reference fields for a service.

4) Network Information Table (NIT):

the location of the NIT is defined in the present document in compliance with ISO/IEC 13818-1 [1] specification, but the data format is outside the scope of ISO/IEC 13818-1 [1]. It is intended to provide information about the physical network. The syntax and semantics of the NIT are defined in the present document.

5) Satellite Access Table (SAT):

the SAT gives additional information on the physical satellite network. The syntax and semantics of the SAT are defined in the present document.

In addition to the PSI, data is needed to provide identification of services and events for the user. The coding of this data is defined in the present document. In contrast with the PAT, CAT, and PMT of the PSI, which give information only for the multiplex in which they are contained (the actual multiplex), the additional information defined within the present document can also provide information on services and events carried by different multiplexes, and even on other networks. This data is structured as nine tables:

1) Bouquet Association Table (BAT):

the BAT provides information regarding bouquets. As well as giving the name of the bouquet, it provides a list of services for each bouquet.

2) Service Description Table (SDT):

the SDT contains data describing the services in the system e.g. names of services, the service provider, etc.

3) Event Information Table (EIT):

- the EIT contains data concerning events or programmes such as event name, start time, duration, etc.;

- the use of different descriptors allows the transmission of different kinds of event information e.g. for different service types.

4) Running Status Table (RST):

the RST gives the status of an event (running/not running). The RST updates this information and allows timely automatic switching to events.

5) Time and Date Table (TDT):

the TDT gives information relating to the present time and date. This information is given in a separate table due to the frequent updating of this information.

6) Time Offset Table (TOT):

the TOT gives information relating to the present time and date and local time offset. This information is given in a separate table due to the frequent updating of the time information.

7) Stuffing Table (ST):

the ST is used to invalidate existing sections, for example at delivery system boundaries.

8) Selection Information Table (SIT):

the SIT is used only in "partial" (i.e. recorded) bitstreams. It carries a summary of the SI information required to describe the streams in the partial bitstream.

9) Discontinuity Information Table (DIT):

the DIT is used only in "partial" (i.e. recorded) bitstreams. It is inserted where the SI information in the partial bitstream may be discontinuous.

Where applicable the use of descriptors allows a flexible approach to the organization of the tables and allows for future compatible extensions.
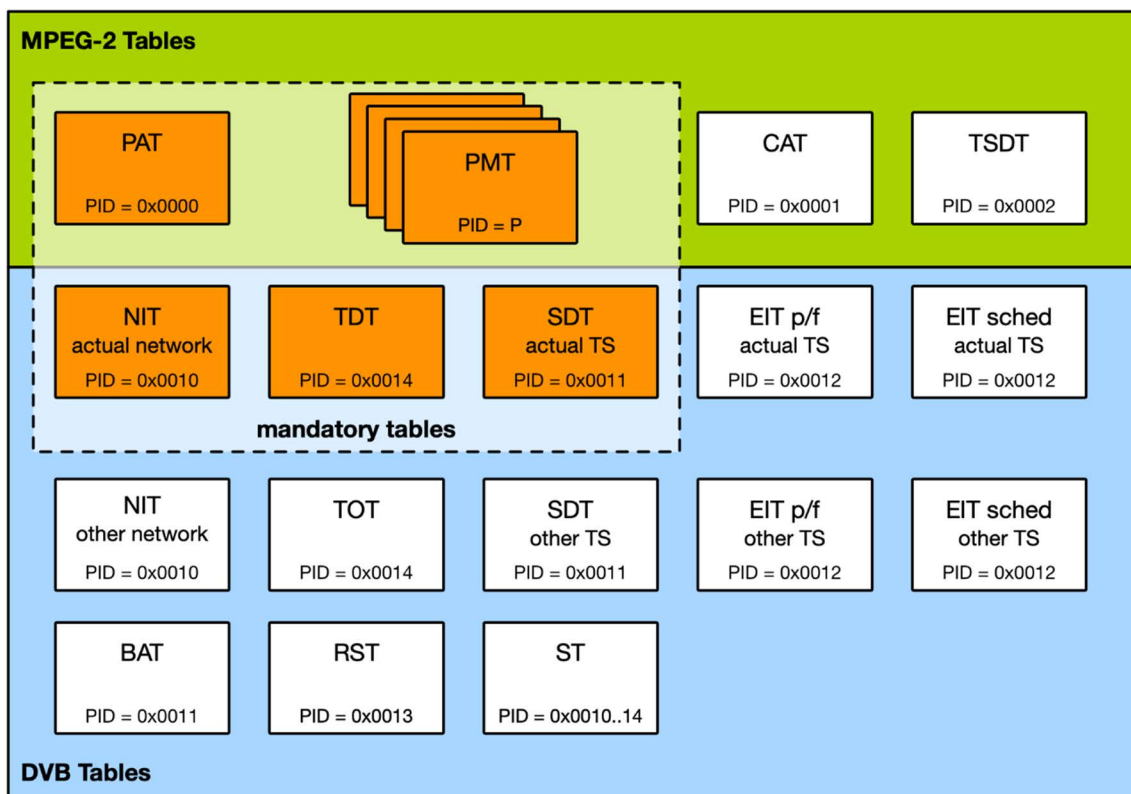


**Figure 2: General organization of the SI**

# 5 Service information tables

## 5.1 Service information table mechanism

### 5.1.1 Use of table sections

The SI specified in the present document and MPEG-2 PSI tables shall be segmented into one or more sections before being inserted into TS packets.

The tables listed in clause 4 are conceptual in that they need never be regenerated in a specified form within an IRD. The tables, when transmitted shall not be scrambled, with the exception of the EIT, which may be scrambled if required (see clause 5.1.5).

A section is a syntactic structure that shall be used for mapping all MPEG-2 tables and SI tables specified in the present document, into TS packets.

These SI syntactic structures conform to the private section syntax defined in ISO/IEC 13818-1 [1].

Sections may be variable in length. The sections within each table are limited to 1 024 bytes in length, except for sections within the EIT and SAT which are limited to 4 096 bytes. Each section is uniquely identified by the combination of the following elements:

1)  *table_id*:

    -   The *table_id* identifies to which table the section belongs.

    -   Some *table_id*s have been defined by ISO and others by ETSI. Other values of the *table_id* can be allocated by the user for private purposes. The list of values of *table_id* is contained in table 2.

2)  *table_id_extension*:

   - The *table_id_extension* is used for identification of a sub_table.

   - The interpretation of each sub_table is given in clause 5.2.

3)  *section_number*:

   - The *section_number* field allows the sections of a particular sub_table to be reassembled in their original order by the decoder. It is recommended, that sections are transmitted in numerical order, unless it is desired to transmit some sections of the sub_table more frequently than others, e.g. due to random access considerations.

   - For the SI tables as specified in the present document, section numbering applies to sub_tables.

4)  *version_number*:

   - When the characteristics of the TS described in the SI given in the present document change (e.g. new events start, different composition of elementary streams for a given service), then new SI data shall be sent containing the updated information. A new version of the SI data is signalled by sending a sub_table with the same identifiers as the previous sub_table containing the relevant data, but with the next value of *version_number*.

   - For the SI tables specified in the present document, the *version_number* applies to all sections of a sub_table.

5)  *current_next_indicator*:

   - Each section shall be numbered as valid "now" (current), or as valid in the immediate future (next). This allows the transmission of a future version of the SI in advance of the change, giving the decoder the opportunity to prepare for the change. There is however, no requirement to transmit the next version of a section in advance, but if it is transmitted, then it shall be the next correct version of that section.

## 5.1.2     Mapping of sections into DVB transport stream packets

Sections shall be mapped directly into TS packets. Sections may start at the beginning of the payload of a TS packet, but this is not a requirement, because the start of the first section in the payload of a TS packet is pointed to by the *pointer_field*. There is never more than one *pointer_field* in a TS packet, as the start of any other section can be identified by counting the length of the first and any subsequent sections, since no gaps between sections within a TS packet are allowed by the syntax.

Within TS packets of any single *PID* value, one section is finished before the next one is allowed to be started, or else it is not possible to identify to which section header the data belongs. If a section finishes before the end of a TS packet, but it is not convenient to open another section, a stuffing mechanism may be used to fill up the space.

Stuffing may be performed by filling each remaining byte of the TS packet with the value 0xFF. Consequently the value 0xFF shall not be used for the *table_id*. If the byte immediately following the last byte of a section takes the value of 0xFF, then the rest of the TS packet shall be stuffed with 0xFF bytes. These bytes may be discarded by a decoder. Stuffing may also be performed using the *adaptation_field* mechanism.

For a more detailed description of the mechanism and functionality, specifically refer to clause 2.4.4 and annex C of ISO/IEC 13818-1 [1].

## 5.1.3     Coding of PID and table_id fields

Table 1 lists the *PID* values which shall be used for the TS packets which carry SI sections.

**Table 1: PID allocation for SI**

| Table | PID value |
|---|---|
| PAT | 0x0000 |
| CAT | 0x0001 |
| Transport Stream Description Table (TSDT) | 0x0002 |
| reserved | 0x0003 to 0x000F |
| NIT, ST | 0x0010 |
| SDT, BAT, ST | 0x0011 |
| EIT, ST, Content Identifier Table (CIT) (ETSI TS 102 323 [21]) | 0x0012 |
| RST, ST | 0x0013 |
| TDT, TOT, ST | 0x0014 |
| network synchronization | 0x0015 |
| Resolution Authority Record Notification Table (RNT) (ETSI TS 102 323 [21]) | 0x0016 |
| reserved for future use | 0x0017 to 0x001A |
| SAT | 0x001B |
| link-local inband signalling | 0x001C |
| measurement | 0x001D |
| DIT | 0x001E |
| SIT | 0x001F |

The PID value 0x001C allocated to link-local inband signalling shall not be used on any broadcast signals. It shall only be used between devices in a controlled environment.

   NOTE:    The PID 0x001C can for example be used within a broadcast centre, between a receiver device and a
            Conditional Access Module (CAM), or on private satellite links.

Table 2 lists the values which shall be used for *table_id* for the service information, defined in the present document.

**Table 2: Allocation of table_id values**

| *table_id* | Description |
|---|---|
| 0x00 | `program_association_section` |
| 0x01 | `conditional_access_section` |
| 0x02 | `program_map_section` |
| 0x03 | `transport_stream_description_section` |
| 0x04 to 0x3F | reserved |
| 0x40 | `network_information_section` - actual network |
| 0x41 | `network_information_section` - other network |
| 0x42 | `service_description_section` - actual DVB transport stream |
| 0x43 to 0x45 | reserved for future use |
| 0x46 | `service_description_section` - other DVB transport stream |
| 0x47 to 0x49 | reserved for future use |
| 0x4A | `bouquet_association_section` |
| 0x4B | update notification table section (ETSI TS 102 006 [20]) |
| 0x4C | IP/MAC_notification_section (ETSI EN 301 192 [3] - see note 2) |
| 0x4D | `satellite_access_section` |
| 0x4E | `event_information_section` - actual DVB transport stream, present/following |
| 0x4F | `event_information_section` - other DVB transport stream, present/following |
| 0x50 to 0x5F | `event_information_section` - actual DVB transport stream, schedule |
| 0x60 to 0x6F | `event_information_section` - other DVB transport stream, schedule |
| 0x70 | `time_date_section` |
| 0x71 | `running_status_section` |
| 0x72 | `stuffing_section` |
| 0x73 | `time_offset_section` |
| 0x74 | application information section (ETSI TS 102 812 [26]) |
| 0x75 | container section (ETSI TS 102 323 [21]) |
| 0x76 | related content section (ETSI TS 102 323 [21]) |
| 0x77 | content identifier section (ETSI TS 102 323 [21]) |
| 0x78 | MPE-FEC section (ETSI EN 301 192 [3]) |
| 0x79 | resolution provider notification section (ETSI TS 102 323 [21]) |
| 0x7A | MPE-IFEC section (ETSI TS 102 772 [23]) |
| 0x7B | protection message section (ETSI TS 102 809 [25]) |
| 0x7C | downloadable font info section (ETSI EN 303 560 [12] - see note 2) |
| 0x7D | reserved for future use |
| 0x7E | `discontinuity_information_section` |
| 0x7F | `selection_information_section` |
| 0x80 to 0xFE | user defined |
| 0xFF | reserved (see note 1) |
| NOTE 1: See clause C.4 of ISO/IEC 13818-1 [1] for why *table_id* of 0xFF can never be used. | |
| NOTE 2: *table_id* 0x4C was previously accidentally assigned to both of these two DVB specifications, this has now been corrected. | |

## 5.1.4 Repetition rates and random access

### 5.1.4.1 Rates for DVB PSI and SI

In systems where acquisition time of PSI and SI in DVB transport streams is important, it is recommended to continuously re-transmit these sections at regular intervals, even when no changes occur. Clause 4.4 of ETSI TS 101 211 [i.1] makes recommendations for how often PSI and SI sections should be re-transmitted.

For SI specified within the present document the minimum time interval between the arrival of the last byte of a section to the first byte of the next transmitted section with the same PID, *table_id* and *table_id_extension* and with the same or different *section_number* shall be 25 ms. This limit applies for TSs with a total data rate of up to 100 Mbit/s.

NOTE: These requirements do not apply to SAT.

### 5.1.4.2 Rates for Satellite Access Tables

Clause 4.4 of ETSI TS 101 211 [i.1] makes recommendations for how often the SAT should be re-transmitted.

## 5.1.5 Scrambling

With the exception of the EIT carrying schedule information, all tables specified in the present document shall not be scrambled. One method for scrambling the EIT schedule table is given in annex E. If a scrambling method operating over TS packets is used, it may be necessary to use a stuffing mechanism to fill from the end of a section to the end of a packet so that any transitions between scrambled and unscrambled data occur at packet boundaries.

In order to identify the CA streams which control the descrambling of the EIT data, a scrambled EIT schedule table shall be identified in the PSI. *service_id* value 0xFFFF is allocated to identifying a scrambled EIT, and the program map section for this service shall describe the EIT as a private stream and shall include one or more `CA_descriptor` (defined in ISO/IEC 13818-1 [1]) which give the PID values and optionally, other private data to identify the associated CA streams. *service_id* value 0xFFFF shall not be used for any other service.

## 5.1.6 Bit order and transmission order

The present clause defines the bit order for encoding unsigned integer (unsigned integer most significant bit first (uimsbf)), bit string (bit string left bit first (bslbf)), two's complement integer (two's complement integer, most significant (sign) bit first (tcimsbf)), and remainder polynomial coefficients (remainder polynomial coefficients highest order first (rpchof)) fields in tables (see clause 5) and descriptors (see clause 6), as well as the order in which these fields are to be transmitted. The present clause also explains and illustrates these provisions.

Within fields, the convention for assigning bit positions -- the bit order - follows the big-endian convention, i.e. more significant bits are assigned positions with higher index values. This results in the most significant bit (msb) being assigned the highest index position, and the least significant bit (lsb) being assigned the lowest index position within a field. Since, by convention, the indices of bit positions count upwards starting from zero, the least significant bit appears at index position zero, denoted as "$b_0$". Consequently, when the field is N bits wide, the most significant bit appears at index position N-1. These principles equally apply to all types of fields. Some example fields are shown in figure 3.
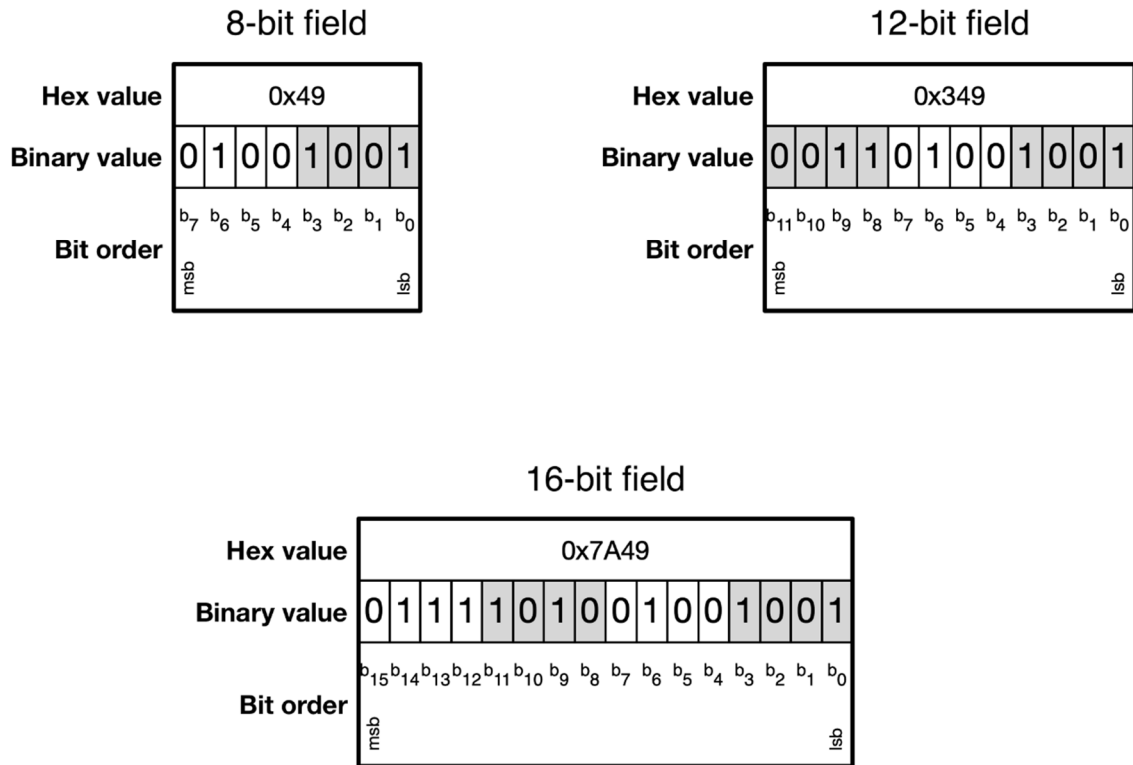
**Figure 3: Examples of bit order within fields**

To define useful elements for conveying service information, the present document defines higher level data structures, like e.g. tables and descriptors, by concatenating various fields of given width and semantics. When such a data structure is to be conveyed to IRDs, it is transmitted over a broadcast link using one of the DVB physical layer standards (e.g. DVB Terrestrial Framing and Modulation (DVB-T), DVB Satellite Framing and Modulation (DVB-S), or DVB Cable Framing and Modulation (DVB-C)). Since all DVB physical layers are bit pipes, any data needs to be serialized into a bit stream before transmission. This serialization follows two rules:

1)    The fields comprising a data structure shall be transmitted in the order they appear in the structure's syntax definition, top to bottom.

2)    Within each field, the bits shall be transmitted in descending order of index position.

An example for applying these rules is shown in figure 4. Part (a) of figure 4 illustrates rule 1), and shows the syntax definition for the `country_availability_descriptor` (see clause 6.2.10). The red arrow to the left of the syntax definition table indicates the top to bottom sequence of transmission of the fields. Part (b) of figure 4 illustrates rule 2), and shows how the bits comprising the fields are serialized. Again, the red arrow below the serialization indicates the sequence of transmission in descending order of index position. For reference, some example values for the fields have been chosen, and their hexadecimal representation is also given.

**Figure 4: Example of transmission order**

Based on these rules, the names "unsigned integer, most significant bit first" (uimsbf) for unsigned integer fields, and "bit string, left bit first" (bslbf) for bit string fields are conveniently explained.

Unsigned integer fields represent numbers within a polyadic number system, which are composed of digits and where the position of each digit indicates to which power of the number system's base the digit refers. Due to the use of the big-endian convention for the bit order within each field, the most significant bit (msb) of an N-bit unsigned integer field is assigned the highest possible index N-1. Following rule 2) (transmission in descending order of index position), the most significant bit of a field is transmitted first, and the least significant bit is transmitted last. Hence the name of "unsigned integer, most significant bit first" (uimsbf).

Bit string fields - as opposed to unsigned integer fields - do not represent numbers, and hence their bit index positions are not associated with higher or lower significance of the bits, but only serve to uniquely identify the bits within a field. To still allow unambiguous serialization and deserialization of bit string fields, the provisions of the present document use the bit index positions $b_0$, $b_1$, $b_2$, etc. to define the encoding and semantics of each bit in a bit string field. When all bits of a bit string field are rendered in printing according to the big-endian convention, the bit with the highest index position N-1 appears on the left, and the bit with the lowest index position zero appears on the right. Following rule 2) (transmission in descending order of index position), the bit with the highest index position (appearing on the left) is transmitted first. Hence the name of "bit string, left bit first" (bslbf).

# 5.2 Table definitions

## 5.2.0 Introduction

The following clauses describe the syntax and semantics of the different types of table.

NOTE:    The symbols and abbreviations, and the method of describing syntax used in the present document are the same as those defined in clause 2.2 and clause 2.3 of ISO/IEC 13818-1 [1].

## 5.2.1    Network Information Table

The NIT (see table 3) conveys information relating to the physical organization of the multiplexes/TSs carried via a given network, and the characteristics of the network itself. The combination of *original_network_id* and *transport_stream_id* allow each TS to be uniquely identified throughout the application area of the present document. Networks are assigned individual *network_id* values, which serve as unique identification codes for networks. The allocation of these codes is specified in ETSI TS 101 162 [15]. The *network_id* and the *original_network_id* can take the same value, or may have to take different values subject to the allocation constraints for *original_network_id* and *network_id* as are defined in ETSI TS 101 162 [15].

Guidelines for the processing of SI at transitions between delivery system boundaries, e.g. from satellite to cable or Satellite Master Antenna TeleVision (SMATV) systems, can be found in ETSI TS 101 211 [i.1].

IRDs may be able to store the NIT information in non-volatile memory in order to minimize the access time when switching between channels ("channel hopping"). It is also possible to transmit a NIT for other networks in addition to the actual network. Differentiation between the NIT for the actual network and the NIT for other networks is achieved using different *table_id* values (see table 2).

The NIT shall be segmented into `network_information_section` using the syntax of table 3. Any sections forming part of an NIT shall be transmitted in TS packets with a PID value of 0x0010. Any sections of the NIT which describe the actual network (that is, the network of which the TS containing the NIT is a part) shall have the *table_id* 0x40 with the same *table_id_extension* (*network_id*). The *network_id* field shall take the value assigned to the actual network according to ETSI TS 101 162 [15]. Any sections of a NIT which refer to a network other than the actual network shall take a *table_id* value of 0x41 and the *network_id* shall take the value assigned to the other network according to ETSI TS 101 162 [15].

**Table 3: Network information section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| network_information_section() { |  |  |
|     table_id | 8 | uimsbf |
|     section_syntax_indicator | 1 | bslbf |
|     reserved_future_use | 1 | bslbf |
|     reserved | 2 | bslbf |
|     section_length | 12 | uimsbf |
|     network_id | 16 | uimsbf |
|     reserved | 2 | bslbf |
|     version_number | 5 | uimsbf |
|     current_next_indicator | 1 | bslbf |
|     section_number | 8 | uimsbf |
|     last_section_number | 8 | uimsbf |
|     reserved_future_use | 4 | bslbf |
|     network_descriptors_length | 12 | uimsbf |
|     for (i=0;i<N;i++) { |  |  |
|         descriptor() |  |  |
|     } |  |  |
|     reserved_future_use | 4 | bslbf |
|     transport_stream_loop_length | 12 | uimsbf |
|     for (i=0;i<N;i++) { |  |  |
|         transport_stream_id | 16 | uimsbf |
|         original_network_id | 16 | uimsbf |
|         reserved_future_use | 4 | bslbf |
|         transport_descriptors_length | 12 | uimsbf |
|         for (j=0;j<N;j++) { |  |  |
|             descriptor() |  |  |
|         } |  |  |
|     } |  |  |
|     CRC_32 | 32 | rpchof |
| } |  |  |

Semantics for the network information section:

*table_id***:** See table 2.

*section_syntax_indicator***:** This 1-bit field shall be set to 0b1.

*section_length*: This is a 12-bit field, the first two bits of which shall be 0b00. It specifies the number of bytes of the section, starting immediately following the *section_length* field and including the Cyclic Redundancy Check (CRC). The value in the *section_length* field shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

*network_id*: This is a 16-bit field which serves as a label to identify the delivery system, about which the NIT informs, from any other delivery system. It shall be coded according to ETSI TS 101 162 [15].

*version_number*: This 5-bit field is the version number of the sub_table. The *version_number* shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the *current_next_indicator* is set to 0b1, then the *version_number* shall be that of the currently applicable sub_table. When the *current_next_indicator* is set to 0b0, then the *version_number* shall be that of the next applicable sub_table.

*current_next_indicator*: This 1-bit indicator, when set to 0b1 indicates that the sub_table is the currently applicable sub_table. When the bit is set to 0b0, it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

*section_number*: This 8-bit field gives the number of the section. The *section_number* of the first section in the sub_table shall be 0x00. The *section_number* shall be incremented by 1 with each additional section with the same *table_id* and *network_id*.

*last_section_number*: This 8-bit field specifies the number of the last section (that is, the section with the highest *section_number*) of the sub_table of which this section is part.

*Network_descriptors_length*: This 12-bit field gives the total length in bytes of the following descriptors.

*transport_stream_loop_length*: This is a 12-bit field specifying the total length in bytes of the TS loops that follow, ending immediately before the first *CRC_32* byte.

*transport_stream_id*: This is a 16-bit field which serves as a label for identification of this TS from any other multiplex within the delivery system.

*original_network_id*: This 16-bit field gives the label identifying the *network_id* of the originating delivery system.

*transport_descriptors_length*: This is a 12-bit field specifying the total length in bytes of TS descriptors that follow.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

## 5.2.2    Bouquet Association Table

The BAT (see table 4) provides information regarding bouquets. A bouquet is a collection of services, which may traverse the boundary of a network.

The BAT shall be segmented into `bouquet_association_sections` using the syntax of table 4. Any section forming part of a BAT shall be transmitted in TS packets with a PID value of 0x0011. The sections of a BAT sub_table describing a particular bouquet shall have the *bouquet_id* field taking the value assigned to the bouquet according to ETSI TS 101 162 [15]. All BAT sections shall take a *table_id* value of 0x4A.

**Table 4: Bouquet association section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| bouquet_association_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    bouquet_id | 16 | uimsbf |
|    reserved | 2 | bslbf |
|    version_number | 5 | uimsbf |
|    current_next_indicator | 1 | bslbf |
|    section_number | 8 | uimsbf |
|    last_section_number | 8 | uimsbf |
|    reserved_future_use | 4 | bslbf |
|    bouquet_descriptors_length | 12 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       descriptor() | | |
|    } | | |
|    reserved_future_use | 4 | bslbf |
|    transport_stream_loop_length | 12 | uimsbf |
|    for(i=0;i<N;i++) { | | |
|       transport_stream_id | 16 | uimsbf |
|       original_network_id | 16 | uimsbf |
|       reserved_future_use | 4 | bslbf |
|       transport_descriptors_length | 12 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          descriptor() | | |
|       } | | |
|    } | | |
|    CRC_32 | 32 | rpchof |
| } | | |

Semantics for the bouquet association section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b1.

*section_length*: This is a 12-bit field, the first two bits of which shall be 0b00. It specifies the number of bytes of the section, starting immediately following the *section_length* field and including the CRC. The value in the *section_length* field shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

*bouquet_id*: This is a 16-bit field which serves as a label to identify the bouquet. It shall be coded according to ETSI TS 101 162 [15].

*version_number*: This 5-bit field is the version number of the sub_table. The *version_number* shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the *current_next_indicator* is set to 0b1, then the *version_number* shall be that of the currently applicable sub_table. When the *current_next_indicator* is set to 0b0, then the *version_number* shall be that of the next applicable sub_table.

*current_next_indicator*: This 1-bit indicator, when set to 0b1 indicates that the sub_table is the currently applicable sub_table. When the bit is set to 0b0, it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

*section_number*: This 8-bit field gives the number of the section. The *section_number* of the first section in the sub_table shall be 0x00. The *section_number* shall be incremented by 1 with each additional section with the same *table_id* and *bouquet_id*.

*last_section_number*: This 8-bit field specifies the number of the last section (that is, the section with the highest *section_number*) of the sub_table of which this section is part.

*bouquet_descriptors_length*: This 12-bit field gives the total length in bytes of the following descriptors.

*transport_stream_loop_length*: This is a 12-bit field specifying the total length in bytes of the TS loops that follow, ending immediately before the first *CRC_32* byte.

*transport_stream_id*: This is a 16-bit field which serves as a label for identification of this TS from any other multiplex within the delivery system.

*original_network_id*: This 16-bit field gives the label identifying the *network_id* of the originating delivery system.

*transport_descriptors_length*: This is a 12-bit field specifying the total length in bytes of TS descriptors that follow.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

## 5.2.3     Service Description Table

Each sub_table of the SDT (see table 5) shall describe services that are contained within a particular TS. The services may be part of the actual TS or part of other TSs, these being identified by means of the *table_id* (see table 2).

The SDT shall be segmented into service_description_sections using the syntax of table 5. Any sections forming part of an SDT shall be transmitted in TS packets with a PID value of 0x0011. Any sections of the SDT which describe the actual TS (that is, the TS containing the SDT) shall have the *table_id* value 0x42 with the same *table_id_extension* (*transport_stream_id*) and with the same *original_network_id*. Any sections of an SDT which refer to a TS other than the actual TS shall take a *table_id* value of 0x46.

**Table 5: Service description section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_description_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    transport_stream_id | 16 | uimsbf |
|    reserved | 2 | bslbf |
|    version_number | 5 | uimsbf |
|    current_next_indicator | 1 | bslbf |
|    section_number | 8 | uimsbf |
|    last_section_number | 8 | uimsbf |
|    original_network_id | 16 | uimsbf |
|    reserved_future_use | 8 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       service_id | 16 | uimsbf |
|       reserved_future_use | 6 | bslbf |
|       EIT_schedule_flag | 1 | bslbf |
|       EIT_present_following_flag | 1 | bslbf |
|       running_status | 3 | uimsbf |
|       free_CA_mode | 1 | bslbf |
|       descriptors_length | 12 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          descriptor() | | |
|       } | | |
|    } | | |
|    CRC_32 | 32 | rpchof |
| } | | |

Semantics for the service description section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b1.

*section_length*: This is a 12-bit field, the first two bits of which shall be 0b00. It specifies the number of bytes of the section, starting immediately following the *section_length* field and including the CRC. The value in the *section_length* field shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

*transport_stream_id*: This is a 16-bit field which serves as a label for identification of this TS from any other multiplex within the delivery system.

*version_number*: This 5-bit field is the version number of the sub_table. The *version_number* shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the *current_next_indicator* is set to 0b1, then the *version_number* shall be that of the currently applicable sub_table. When the *current_next_indicator* is set to 0b0, then the *version_number* shall be that of the next applicable sub_table.

*current_next_indicator*: This 1-bit indicator, when set to 0b1 indicates that the sub_table is the currently applicable sub_table. When the bit is set to 0b0, it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

*section_number*: This 8-bit field gives the number of the section. The *section_number* of the first section in the sub_table shall be 0x00. The *section_number* shall be incremented by 1 with each additional section with the same *table_id*, *transport_stream_id*, and *original_network_id*.

*last_section_number*: This 8-bit field specifies the number of the last section (that is, the section with the highest *section_number*) of the sub_table of which this section is part.

*original_network_id*: This 16-bit field gives the label identifying the *network_id* of the originating delivery system.

*service_id*: This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding program_map_section, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (Near Video On Demand (NVOD) reference services) the *service_id* does not have a corresponding *program_number*.

*EIT_schedule_flag*: This is a 1-bit field which when set to 0b1 indicates that EIT schedule information for the service is present in the current TS (see ETSI TS 101 211 [i.1] for information on maximum time interval between occurrences of an EIT schedule sub_table). If the flag is set to 0b0 then the EIT schedule information for the service should not be present in the TS.

*EIT_present_following_flag*: This is a 1-bit field which when set to 0b1 indicates that EIT present/following information for the service is present in the current TS (see ETSI TS 101 211 [i.1] for information on maximum time interval between occurrences of an EIT present/following sub_table). If the flag is set to 0b0 then the EIT present/following information for the service should not be present in the TS.

*running_status*: This is a 3-bit field indicating the status of the service as defined in table 6.

**Table 6: Running status**

| running_status | Description |
|---|---|
| 0 | undefined |
| 1 | not running |
| 2 | starts in a few seconds (e.g. for video recording) |
| 3 | pausing |
| 4 | running |
| 5 | service off-air |
| 6 to 7 | reserved for future use |

For an NVOD reference service the value of the *running_status* shall be set to 0.

*free_CA_mode*: This 1-bit field, when set to 0b0 indicates that all the component streams of the service are not scrambled. When set to 0b1 it indicates that access to one or more streams may be controlled by a conditional access system.

*descriptors_length*: This 12-bit field gives the total length in bytes of the following descriptors.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

## 5.2.4     Event Information Table

The EIT (see table 7) provides information in chronological order regarding the events contained within each service. Four classifications of EIT have been identified, distinguishable by the use of different *table_id*s (see table 2):

   1)     actual TS, present/following event information = *table_id* = 0x4E;

   2)     other TS, present/following event information = *table_id* = 0x4F;

   3)     actual TS, event schedule information = *table_id* = 0x50 to 0x5F;

   4)     other TS, event schedule information = *table_id* = 0x60 to 0x6F.

All EIT sub_tables for the actual DVB transport stream shall have the same *transport_stream_id* and *original_network_id* values.

The EIT present/following table shall contain only information pertaining to the present event and the chronologically following event carried by a given service on either the actual TS or another TS, except in the case of a NVOD reference service where it may have more than two event descriptions. The EIT present/following table is optional. Its presence or absence shall be signalled by setting the *EIT_present_following_flag* in the SDT.

The event schedule tables for either the actual TS or other TSs, contain a list of events, in the form of a schedule including events other than the present and following events. The EIT schedule tables are optional. Their presence or absence shall be signalled by setting the *EIT_schedule_flag* in the SDT. The event information shall be chronologically ordered.

The EIT shall be segmented into `event_information_sections` using the syntax of table 7. Any sections forming part of an EIT shall be transmitted in TS packets with a PID value of 0x0012.

**Table 7: Event information section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| event_information_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    service_id | 16 | uimsbf |
|    reserved | 2 | bslbf |
|    version_number | 5 | uimsbf |
|    current_next_indicator | 1 | bslbf |
|    section_number | 8 | uimsbf |
|    last_section_number | 8 | uimsbf |
|    transport_stream_id | 16 | uimsbf |
|    original_network_id | 16 | uimsbf |
|    segment_last_section_number | 8 | uimsbf |
|    last_table_id | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       event_id | 16 | uimsbf |
|       start_time | 40 | bslbf |
|       duration | 24 | uimsbf |
|       running_status | 3 | uimsbf |
|       free_CA_mode | 1 | bslbf |
|       descriptors_length | 12 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          descriptor() | | |
|       } | | |
|    } | | |
|    CRC_32 | 32 | rpchof |
| } | | |

Semantics for the event information section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b1.

*section_length*: This is a 12-bit field. It specifies the number of bytes of the section, starting immediately following the *section_length* field and including the CRC. The *section_length* shall not exceed 4 093 so that the entire section has a maximum length of 4 096 bytes.

*service_id*: This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding program_map_section, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (NVOD reference services) the *service_id* does not have a corresponding *program_number*.

*version_number*: This 5-bit field is the version number of the sub_table. The *version_number* shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the *current_next_indicator* is set to 0b1, then the *version_number* shall be that of the currently applicable sub_table. When the *current_next_indicator* is set to 0b0, then the *version_number* shall be that of the next applicable sub_table.

*current_next_indicator*: This 1-bit indicator, when set to 0b1 indicates that the sub_table is the currently applicable sub_table. When the bit is set to 0b0, it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

*section_number*: This 8-bit field gives the number of the section. The *section_number* of the first section in the sub_table shall be 0x00. The *section_number* shall be incremented by 1 with each additional section with the same *table_id*, *service_id*, *transport_stream_id*, and *original_network_id*. In this case, the sub_table may be structured as a number of segments. Within each segment the *section_number* shall increment by 1 with each additional section, but a gap in numbering is permitted between the last section of a segment and the first section of the adjacent segment.

*last_section_number*: This 8-bit field specifies the number of the last section (that is, the section with the highest *section_number*) of the sub_table of which this section is part.

*transport_stream_id*: This is a 16-bit field which serves as a label for identification of this TS from any other multiplex within the delivery system.

*original_network_id*: This 16-bit field gives the label identifying the *network_id* of the originating delivery system.

*segment_last_section_number*: This 8-bit field specifies the number of the last section of this segment of the sub_table. For sub_tables which are not segmented, this field shall be set to the same value as the *last_section_number* field.

*last_table_id*: This 8-bit field identifies the last *table_id* used (see table 2). For EIT present/following tables, this field shall be set to the same value as the *table_id* field. For EIT schedule tables with *table_id* in the range 0x50 to 0x5F, this field shall be set to the largest *table_id* transmitted in this range for this service. For EIT schedule tables with *table_id* in the range 0x60 to 0x6F, this field shall be set to the largest *table_id* transmitted in this range for this service.

NOTE:    This implies that the value of *last_table_id* may be different for each service.

EXAMPLE 1:    Table 7a shows some examples of *last_table_id* values for two services.

**Table 7a: Examples of last_table_id values**

| Transmitted EIT *table_id* | Service | *last_table_id* |
|---|---|---|
| 0x4E | A | 0x4E |
| 0x4F | A | 0x4F |
| 0x50, 0x51 | A | 0x51 |
| 0x60, 0x61, 0x62 | A | 0x62 |
| 0x4E | B | 0x4E |
| 0x4F | B | 0x4F |
| 0x50, 0x51, 0x52, 0x53 | B | 0x53 |
| 0x60 | B | 0x60 |

*event_id*: This 16-bit field contains the identification number of the described event (uniquely allocated within a service definition).

*start_time*: This 40-bit field contains the start time of the event in Universal Time, Co-ordinated (UTC) and Modified Julian Date (MJD) (see annex C). This field is coded as 16 bits giving the 16 lsb of MJD followed by 24 bits coded as 6 digits in 4-bit Binary Coded Decimal (BCD). If the start time is undefined (e.g. for an event in a NVOD reference service) all bits of the field are set to 0b1.

EXAMPLE 2:     93/10/13 12:45:00 is coded as 0xC0 7912 4500.

*duration*: This 24-bit field indicates the duration of the event in hours, minutes, and seconds coded as 6 digits in 4-bit BCD.

EXAMPLE 3:     01:45:30 is coded as 0x01 4530.

*running_status*: This is a 3-bit field indicating the status of the event as defined in table 6. For an NVOD reference event the value of the *running_status* shall be set to 0b0.

*free_CA_mode*: This 1-bit field, when set to 0b0 indicates that all the component streams of the event are not scrambled. When set to 0b1 it indicates that access to one or more streams may be controlled by a conditional access system.

*descriptors_length*: This 12-bit field gives the total length in bytes of the following descriptors.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

## 5.2.5     Time and Date Table

The TDT (see table 8) carries only the UTC time and date information.

The TDT shall consist of a single section using the syntax of table 8. This TDT section shall be transmitted in TS packets with a PID value of 0x0014, and the *table_id* shall take the value 0x70.

**Table 8: Time and date section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| time_date_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    UTC_time | 40 | bslbf |
| } | | |

Semantics for the time and date section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b0.

*section_length*: This is a 12-bit field, the first two bits of which shall be 0b00. It specifies the number of bytes of the section, starting immediately following the *section_length* field and up to the end of the section.

*UTC_time*: This 40-bit field contains the current time and date in UTC and MJD (see annex C). This field is coded as 16 bits giving the 16 lsb of MJD followed by 24 bits coded as 6 digits in 4-bit BCD.

EXAMPLE:        93/10/13 12:45:00 is coded as 0xC0 7912 4500.

## 5.2.6    Time Offset Table

The TOT (see table 9) carries the UTC time and date information and local time offset. The TOT shall consist of a single section using the syntax of table 9. This TOT section shall be transmitted in TS packets with a PID value of 0x0014, and the *table_id* shall take the value 0x73.

**Table 9: Time offset section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| time_offset_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    UTC_time | 40 | bslbf |
|    reserved | 4 | bslbf |
|    descriptors_length | 12 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       descriptor() | | |
|    } | | |
|    CRC_32 | 32 | rpchof |
| } | | |

Semantics for the time offset section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b0.

*section_length*: This is a 12-bit field, the first two bits of which shall be 0b00. It specifies the number of bytes of the section, starting immediately following the *section_length* field and up to the end of the section.

*UTC_time*: This 40-bit field contains the current time and date in UTC and MJD (see annex C). This field is coded as 16 bits giving the 16 lsb of MJD followed by 24 bits coded as 6 digits in 4-bit BCD.

    EXAMPLE:        93/10/13 12:45:00 is coded as 0xC0 7912 4500.

*descriptors_length*: This 12-bit field gives the total length in bytes of the following descriptors.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

## 5.2.7    Running Status Table

The RST (see table 10) allows accurate and rapid updating of the timing status of one or more events. This may be necessary when an event starts early or late due to scheduling changes. The use of a separate table enables fast updating mechanism to be achieved.

The RST shall be segmented into running_status_sections using the syntax of table 10. Any sections forming part of an RST shall be transmitted in TS packets with a PID value of 0x0013, and the *table_id* shall take the value 0x71.

**Table 10: Running status section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| running_status_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       transport_stream_id | 16 | |
|       original_network_id | 16 | uimsbf |
|       service_id | 16 | uimsbf |
|       event_id | 16 | uimsbf |
|       reserved_future_use | 5 | bslbf |
|       running_status | 3 | uimsbf |
|    } | | |
| } | | |

Semantics for the running status section:

*table_id***:** See table 2.

*section_syntax_indicator***:** This 1-bit field shall be set to 0b0.

*section_length***:** This is a 12-bit field, the first two bits of which shall be 0b00. It specifies the number of bytes of the section, starting immediately following the *section_length* field and up to the end of the section. The *section_length* shall not exceed 1 021 so that the entire section has a maximum length of 1 024 bytes.

*transport_stream_id***:** This is a 16-bit field which serves as a label for identification of this TS from any other multiplex within the delivery system.

*original_network_id***:** This 16-bit field gives the label identifying the *network_id* of the originating delivery system.

*service_id***:** This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding program_map_section, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (NVOD reference services) the *service_id* does not have a corresponding *program_number*.

*event_id***:** This 16-bit field contains the identification number of the related event.

*running_status***:** This is a 3-bit field indicating the status of the event as defined in table 6.

## 5.2.8    Stuffing Table

The purpose of this clause (see table 11) is to invalidate existing sections at a delivery system boundary e.g. at a cable head-end. When one section of a sub_table is overwritten, then all the sections of that sub_table shall also be overwritten (stuffed) in order to retain the integrity of the *section_number* field.

**Table 11: Stuffing section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| stuffing_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       data_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the stuffing section:

*table_id***:** See table 2.

*section_syntax_indicator***:** This 1-bit field may take either the value 0b1 or 0b0.

*section_length***:** This is a 12-bit field. It specifies the number of bytes of the section, starting immediately following the *section_length* field and up to the end of the section. The value in the *section_length* field shall not exceed 4 093 so that the entire section has a maximum length of 4 096 bytes.

*data_byte***:** This 8-bit field may take any value and has no meaning.

## 5.2.9      Discontinuity Information Table

See clause 7.1.1.

## 5.2.10     Selection Information Table

See clause 7.1.2.

## 5.2.11     Satellite Access Tables

### 5.2.11.1      Definition

The SAT (see table 11a) convey information in several sub_tables relating to the physical organization of the carriage of satellite delivery systems over satellite. The following clauses describe the syntax and semantics of the different sub_tables. The sub_tables shall be segmented into sections using the private section syntax defined in ISO/IEC 13818-1 [1]. Any sections forming part of a SAT shall be transmitted in TS packets with a PID value of 0x001B.

**Table 11a: Satellite access section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| satellite_access_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    private_indicator | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    satellite_table_id | 6 | uimsbf |
|    table_count | 10 | uimsbf |
|    reserved | 2 | bslbf |
|    version_number | 5 | uimsbf |
|    current_next_indicator | 1 | bslbf |
|    section_number | 8 | uimsbf |
|    last_section_number | 8 | uimsbf |
|    reserved_zero_future_use | 8 | bslbf |
|    if (satellite_table_id == 0) { | | |
|       satellite_position_v2_info() | | |
|    } | | |
|    else if (satellite_table_id == 1) { | | |
|       cell_fragment_info() | | |
|    } | | |
|    else if (satellite_table_id == 2) { | | |
|       time_association_info() | | |
|    } | | |
|    else if (satellite_table_id == 3) { | | |
|       beamhopping_time_plan_info() | | |
|    } | | |
|    else if (satellite_table_id == 4) { | | |
|       satellite_position_v3_info() | | |
|    } | | |
|    else { | | |
|       for (i=0;i<N;i++) { | | |
|          reserved_zero_future_use | 8 | bslbf |
|       } | | |
|    } | | |
|    CRC_32 | 32 | rpchof |
| } | | |

Semantics for the satellite access section:

*table_id***:** See table 2.

*section_syntax_indicator***:** This 1-bit field shall be set to 0b1.

*private_indicator:* This 1-bit field shall be set to 0b1.

*section_length***:** This is a 12-bit field. It specifies the number of bytes of the section, starting immediately following the *section_length* field and including the Cyclic Redundancy Check (CRC). The value in the *section_length* field shall not exceed 4 093 so that the entire section has a maximum length of 4 096 bytes.

*satellite_table_id***:** This is a 6-bit field which identifies each table type. The values of satellite_table_id are defined in table 11b).

**Table 11b: Satellite table id coding**

| *satellite_table_id* | Syntax | Defined in |
|---|---|---|
| 0 | satellite_position_v2_info | clause 5.2.11.2 |
| 1 | cell_fragment_info | clause 5.2.11.3 |
| 2 | time_association_info | clause 5.2.11.4 |
| 3 | beamhopping_time_plan_info | clause 5.2.11.5 |
| 4 | satellite_position_v3_info | clause 5.2.11.5 |
| 5 to 63 | reserved for future use | |

*table_count*: This 10-bit field allows different sub_tables to be distinguished. For the Cell Fragment sub_table, the table_count is arbitrary. For the Time Association sub_table, the table_count shall always be set to 0. For the Satellite Position sub_table, the table_count is equal to the 10 most significant bits of the *satellite_id*. For the Beamhopping Time Plan sub_table, the table_count is equal to the 10 most significant bits of the *beamhopping_time_plan_id*.

> NOTE: When the operator chooses to use consecutive *satellite_id* numbers, then they are placed in the same table_count. By spreading over the 10 most significant bits, a spread over multiple sub_tables can be done.

*version_number*: This 5-bit field is the version number of the sub_table. The *version_number* shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the *current_next_indicator* is set to 0b1, then the *version_number* shall be that of the currently applicable sub_table. When the *current_next_indicator* is set to 0b0, then the *version_number* shall be that of the next applicable sub_table.

*current_next_indicator*: This 1-bit indicator, when set to 0b1 indicates that the sub_table is the currently applicable sub_table. When the bit is set to 0b0, it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

*section_number*: This 8-bit field gives the number of the section. The *section_number* of the first section in the sub_table shall be 0x00. The *section_number* shall be incremented by 1 with each additional section with the same *table_id, satellite_table_id* and *table_count*.

*last_section_number*: This 8-bit field specifies the number of the last section (that is, the section with the highest *section_number*) of the sub_table of which this section is part.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

## 5.2.11.2    Satellite Position v2 info

The SAT contains the satellite ephemeris data for DVB-S2Xv2 delivery in the sections with *satellite_table_id* set to 0. It may be updated at regular intervals to compensate for changes in the satellite ephemeris. The `satellite_position_v2_info` (see table  11c) shall contain ephemeris data for those satellites that constitute a part of a particular network. It is based on the Two Line orbital Element (TLE) system defined in American Institute of Aeronautics and Astronautics, AIAA 2006-6753-Rev1 [i.10]. The epoch is represented in UTC time.

**Table 11c: Satellite position v2 info**

| Syntax | Number of bits | Mnemonic |
|---|---|---|
| satellite_position_v2_info() { | | |
|   for (i=1;i<=N;i++) { | | |
|     satellite_id | 24 | |
|     reserved_zero_future_use | 7 | |
|     position_system | 1 | |
|     if (position_system == 0) { | | |
|       orbital position | 16 | bslbf |
|       west_east_flag | 1 | bslbf |
|       reserved_zero_future_use | 7 | bslbf |
|     } | | |
|     if (position_system == 1) { | | |
|       epoch_year | 8 | uimsbf |
|       day_of_the_year | 16 | uimsbf |
|       day_fraction | 32 | spfmsbf |
|       mean_motion_first_derivative | 32 | spfmsbf |
|       mean_motion_second_derivative | 32 | spfmsbf |
|       drag_term | 32 | spfmsbf |
|       inclination | 32 | spfmsbf |
|       right_ascension_of_the_ascending_node | 32 | spfmsbf |
|       eccentricity | 32 | spfmsbf |
|       argument_of_perigree | 32 | spfmsbf |
|       mean_anomaly | 32 | spfmsbf |
|       mean_motion | 32 | spfmsbf |
|     } | | |
|   } | | |
| } | | |

Semantics for the satellite position 2 table info:

*satellite_id*: This is a 24-bit field which serves as a label to identify the satellite that is detailed in this loop.

*position_system*: This is a 1-bit field that identifies the positioning system that is used in this signalling for this satellite. The value 0 can be used for a satellite in geostationary orbit and the value 1 can be used for any earth-orbiting satellite.

*orbital position*: The orbital_position is a 16-bit field giving the 4-bit BCD values specifying 4 characters of the orbital position in degrees where the decimal point occurs after the third character (e.g. 019,2°).

*west_east_flag*: The west_east_flag is a 1-bit field indicating if the satellite position is in the western or eastern part of the orbit. A value 0 indicates the western position and a value 1 indicates the eastern position.

*epoch_year*: This is a 8-bit field which contains the last 2 digits of the epoch year.

*day_of_the_year*: This is a 16-bit field which contains the epoch day of the year.

*day_fraction*: This is a 32-bit field which contains the epoch day fraction.

*mean_motion_first_derivative*: This is a 32-bit field which contains the mean motion derivative divided by 2 in revolutions/day$^2$.

*mean_motion_second_derivative*: This is a 32-bit field which contains the mean motion second derivative divided by 6 in revolutions/day$^3$.

*drag_term*: Ths is a 32-bit field which contains drag term (or radiation pressure coefficient or BSTAR) in 1/EarthRadii.

*inclination*: This is a 32-bit field which contains the angle between the equator and the orbit plane in degrees.

*right_ascension_of_the_ascending_node*: This 32-bit field is the right ascension of the ascension node in degrees.

*eccentricity*: This 32-bit field is a constant defining the shape of the orbit (0=circular, less than 1=elliptical). The value provided is the mean eccentricity.

*argument_of_perigree*: This 32-bit field is the argument of perigee in degrees.

*mean_anomaly*: This 32-bit field is the mean anomaly in degrees.

*mean_motion*: This 32-bit field is the mean number of orbits per day the object completes in revolutions/day.

### 5.2.11.3    Cell Fragment info

The `cell_fragment_info` (see table  11d) describes cell fragments, areas on earth identified by a center (center_lattitude and center_longitude) and *max_distance* from the center, that are associated with none, one or multiple *delivery_system_id*. Cell fragments can both be static or have a slowly varying center and *max_distance*.

It is possible to span the signalling of a cell fragment over multiple consecutive sections within a single *table_count*. In that case, the first occurence shall have the *first_occurence* bit set to 1 and shall include the latitude, longitude and *max_distance*. The following sections shall then continue to signal for this *cell_fragment_id* with *first_occurence* set to 0 and *last_occurence* set to 0. The last section that contains signalling for this *cell_fragment_id* shall set *first_occurence* to 0 and *last_occurence* to 1. There is no link between the *cell_fragment_id* and the *table_count*, it can be placed arbitrarily.

**Table 11d: Cell fragment info**

| Syntax | Number of bits | Mnemonic |
|---|---|---|
| cell_fragment_info(){ | | |
|   for (i=1;i<=N;i++) { | | |
|     cell_fragment_id | 32 | uimsbf |
|     first_occurence | 1 | bsblf |
|     last_occurence | 1 | bsblf |
|     if (first_occurence == 1) { | | |
|       reserved_zero_future_use | 4 | bsblf |
|       center_latitude | 18 | tcimsbf |
|       reserved_zero_future_use | 5 | bsblf |
|       center_longitude | 19 | tcimsbf |
|       max_distance | 24 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|     } else { | | |
|     reserved_zero_future_use | 4 | bsblf |
|     } | | |
|     delivery_system_id_loop_count | 10 | uimsbf |
|     for (j=0;j<delivery_system_id_loop_count;j++) { | | |
|       delivery_system_id | 32 | uimsbf |
|     } | | |
|     reserved_zero_future_use | 6 | bsblf |
|     new_delivery_system_id_loop_count | 10 | uimsbf |
|     for (k=0;k<new_delivery_system_id_loop_count;k++) { | | |
|       new_delivery_system_id | 32 | uimsbf |
|       time_of_application_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       time_of_application_ext | 9 | uimsbf |
|     } | | |
|     reserved_zero_future_use | 6 | bsblf |
|     obsolescent_delivery_system_id_loop_count | 10 | uimsbf |
|     for (l=0;l<obsolescent_delivery_system_id_loop_count;l++) { | | |
|       obsolescent_delivery_system_id | 32 | uimsbf |
|       time_of_obsolescence_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       time_of_obsolescence_ext | 9 | uimsbf |
|     } | | |
|   } | | |
| } | | |

Semantics for the cell fragment info:

*cell_fragment_id*: This 32-bit field indicates the id of this cell_fragment. It can occur multiple time in consecutive sections if the signalling is spanned over multiple sections.

*first_occurence*: This 1-bit field indicates that this section is the first section in a sequence of 1 or more sections that contain the information for the *cell_fragment_id*. When this bit is set to 1, it is the first section of the sequence. When it is set to 0, it is not the first section.

*last_occurence*: This 1-bit field indicates that this section is the last section in a sequence of 1 or more sections that contain the information for the *cell_fragment_id*. When this bit is set to 1, it is the last section of the sequence. When it is set to 0, it is not the last section.

> NOTE 1: When both *first_occurence* and *last_occurence* are set to 1, then the *cell_fragment_id* is fully described in this section. When both *first_occurence* and *last_occurrence* are set to 0, then this section is in a sequence of consecutive sections describing the *cell_fragment_id*.

*center_latitude*: This 18-bit field indicates the current center of the cell fragment on earth in the WGS84 [i.11] datum, in units of 0,001 degrees. Northern latitudes shall be stated as a positive number, southern latitudes as negative. The range is evidently ±90 degrees.

*center_longitude*: This 19-bit field indicates the current center of the cell fragment on earth in the WGS84 [i.11] datum, in units of 0,001 degrees. Eastern longitudes shall be stated as a positive number, western longitudes as negative. The range is evidently ±180 degrees.

*max_distance*: This 24-bit field indicates the maximum distance on the surface of the earth from the center of the cell-fragment in units of 1 meter that is still considered to be part of the cell fragment.

> NOTE 2: Accuracy of 1 % is sufficient, e.g. using haversine formula.

*delivery_system_id_loop_count*: This 10-bit field indicates the number of *delivery_system_id* provided in the loop that follows. This loop contains delivery systems that are currently stably present.

> NOTE 3: When a cell fragment description spans multiple sections, the total number of *delivery_system_id* for the cell fragment can be larger than this single loop count.

*delivery_system_id*: This 32-bit field indicates the *delivery_system_id* of a delivery system that is serving this cell_fragment.

*new_delivery_system_id_loop_count*: This 10-bit field indicates the number of *new_delivery_system_id* provided in the loop that follows. This loop contains delivery systems that will come active at time_of_application.

> NOTE 4: When a cell fragment description spans multiple sections, the total number of *new_delivery_system_id* for the cell fragment can be larger than this single loop count.

*new_delivery_system_id*: This 32-bit field indicates the *delivery_system_id* of a new delivery system that is soon serving this cell fragment.

*time_of_application_base*: This 33-bit field indicates the NCR of time_of_application div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*time_of_application_ext*: This 9-bit field indicates the NCR of time_of_application modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*obsolescent_delivery_system_id_loop_count*: This 10-bit field indicates the number of *obsolescent_delivery_system_id* provided in the loop that follows. This loop contains delivery systems that will become inactive after time_of_obsolescence.

> NOTE 5: When a cell fragment spans multiple sections, the total number of *obsolescent_delivery_system_id* for the cell fragment can be larger than this single loop count.

*obsolescent_delivery_system_id*: This 32-bit field indicates the *delivery_system_id* of a delivery system that is soon not serving anymore this cell_fragment.

*time_of_obsolescence_base*: This 33-bit field indicates the NCR of time_of_obsolescence div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*time_of_obsolescence_ext*: This 9-bit field indicates the NCR of time_of_obsolescence modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

> NOTE 6:   Based on the list of *delivery_system_id* in the cell fragment, the receiver can build up the list of delivery systems that match with the cell fragment and make a list carriers that are servicing this cell fragment from a satelllite with the *satellite_id* field in the carrier.

## 5.2.11.4      Time Association info

The time_association_info (see table 11e) conveys the time association between NCR and UTC. It allows the receiver to translate the received NCRs in UTC time.

The association type 'UTC with leap second info' (see table 11f) shall only be used if the announcement of leap second info is possible on the system at least 11 hours before the leap second event and the announcement is mandatory during these 11 hours before and up to 1 month after the leap second event.

**Table 11e: Time association info**

| Syntax | Number of bits | Mnemonic |
|---|---|---|
| time_association_info(){ | | |
|    association_type | 4 | uimsbf |
|   if (association_type = 1) { | | |
|     leap59 | 1 | bsblf |
|     leap61 | 1 | bsblf |
|     pastleap59 | 1 | bsblf |
|     pastleap61 | 1 | bsblf |
|   } else { | | |
|     reserved_zero_future_use | 4 | bsblf |
|   } | | |
|   ncr_base | 33 | uimsbf |
|   reserved_zero_future_use | 6 | bsblf |
|   ncr_ext | 9 | uimsbf |
|   association_timestamp_seconds | 64 | uimsbf |
|   association_timestamp_nanoseconds | 32 | uimsbf |
| } | | |

Semantics for the time association info:

*association_type*: This 4-bit field indicates how the association_timestamp is to be interpreted. It shall be set according to table  11f.

**Table 11f: Association type coding**

| association_type | Description |
|---|---|
| 0 | UTC without leap second signalling |
| 1 | UTC with leap second signalling |
| 2 to 15 | reserved |

*leap59*: This 1-bit field may be set to 1 to announce that a leap second will be skipped at the end of the quarter of the year to which the association_timestamp belongs. Otherwise, this field shall be set to 0.

*leap61*: This 1-bit field may be set to 1 to announce that a leap second will be added at the end of the quarter of a year to which the association_timestamp belongs. In case that the association_timestamp refers to a time within such an added leap second, and the leap second is already added to the association_timestamp, then this flag shall be set to 0. In all other cases this field shall be set to 0.

*pastleap59*: This 1-bit field may be set to 1 to announce that a leap second was skipped at the end of the quarter of a year previous to the quarter to which the association_timestamp belongs. Otherwise, this field shall be set to 0.

*pastleap61*: This 1-bit field may be set to 1 to announce that a leap second is currently being added, when the association_timestamp refers to the last second in a quarter of a year. This field may also be set to 1 to announce that a leap second was added at the end of the quarter of a year previous to the quarter to which the association_timestamp belongs except when it belongs to the last second in that quarter. Otherwise, this field shall be set to 0.

*ncr_base*: This 33-bit field indicates the NCR time div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1]. The NCR time is associated with the association_timestamp. The NCR time used in the association shall be between 648 000 000 (2 hours) in the past and 7 776 000 000 (24 hours) in the future. Typically, it will be very close to the current NCR.

*ncr_ext*: This 9-bit field indicates the NCR time mod 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*association_timestamp_seconds*: This 64-bit field contains the number of seconds of the association_timestamp since January 1st, 1970 00:00:00.

*association_timestamp_nanoseconds*: This 32-bit field, with a maximum of 1 000 000 000, contains the number of nanoseconds of the association_timestamp on top of the *association_timestamp_seconds* since January 1st, 1970 00:00:00.

## 5.2.11.5 Beamhopping Time Plan info

The `beamhopping_time_plan_info` (see table 11g) conveys a beamhopping time plan, identified by the *beamhopping_time_plan_id* with information relating to the period(s) in time that the beam will illuminate the cell each beamhopping cycle.

At *time_of_application* a cycle will start for the duration of the *cycle_time*, and it will be followed by other cycles with the same illumination time plan. If the *time_of_application* is in the past, then the time plan-shall be considered the current active time plan. If the *time_of_application* is in the future, then that time plan is active at that *time_of_application*.

The NCR time used in the association shall be between 648 000 000 (2 hours) in the past and 7 776 000 000 (24 hours) in the future.

**Table 11g: Beamhopping time plan info**

| Syntax | Number of bits | Mnemonic |
|---|---|---|
| beamhopping_time_plan_info() { | | |
|   for (i=1;i<=N;i++) { | | |
|     beamhopping_time_plan_id | 32 | uimsbf |
|     reserved_zero_future_use | 4 | bsblf |
|     beamhopping_time_plan_length | 12 | uimsbf |
|     reserved_zero_future_use | 6 | bsblf |
|     time_plan_mode | 2 | uimsbf |
|     time_of_application_base | 33 | uimsbf |
|     reserved_zero_future_use | 6 | bsblf |
|     time_of_application_ext | 9 | uimsbf |
|     cycle_duration_base | 33 | uimsbf |
|     reserved_zero_future_use | 6 | bsblf |
|     cycle_duration_ext | 9 | uimsbf |
|     if time_plan_mode == 0 { | | |
|       dwell_duration_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       dwell_duration_ext | 9 | uimsbf |
|       on_time_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       on_time_ext | 9 | uimsbf |
|     } | | |
|     if (time_plan_mode == 1) { | | |
|       reserved_zero_future_use | 1 | bsblf |
|       bit_map_size | 15 | uimsbf |
|       reserved_zero_future_use | 1 | bsblf |
|       current_slot | 15 | uimsbf |
|       for (j=1;j<=bit_map_size;j++) { | | |
|         slot_transmission_on | 1 | bsblf |
|       } | | |
|       for (k=1;k<=J;k++) { | | |
|         padding_bit | 1 | bsblf |
|       } | | |
|     } | | |
|     if (time_plan_mode == 2) { | | |
|       grid_size_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       grid_size_ext | 9 | uimsbf |
|       revisit_duration_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       revisit_duration_ext | 9 | uimsbf |
|       sleep_time_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | uimsbf |
|       sleep_time_ext | 9 | bsblf |
|       sleep_duration_base | 33 | uimsbf |
|       reserved_zero_future_use | 6 | bsblf |
|       sleep_duration_ext | 9 | uimsbf |
|     } | | |
|   } | | |
| } | | |

Semantics for the beamhopping time plan info:

***beamhopping_time_plan_id*****:** This is a 32-bit field which serves as a label to identify the beamhopping time plan that is detailed in this loop.

***beamhopping_time_plan_length*****:** This 12-bit field specifies the number of bytes of this iteration in the loop, starting with the *beamhopping_time_plan_id* and ending at the end of the loop. It allows to quickly jump to the next iteration.

*time_plan_mode*: This 2-bit field indicates the mode that the time plan is communicated in this iteration. When set to 0, then there will be 1 transmission each cycle, starting on the NCR time *on_time_base* base units and *on_time_ext* extension units and lasting *dwell_duration_base* NCR base units and *dwell_duration_ext* NCR extension units. When set to 1, then there can be multiple transmissions in each cycle, specified via a bit map. When set to 2, then the *grid_size* will be signalled, but when the illumination will take place is not signalled.

*time_of_application_base*: This 33-bit field indicates the NCR of time_of_application div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*time_of_application_ext*: This 9-bit field indicates the NCR of time_of_application modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*cycle_duration_base*: This 33-bit field indicates the duration in NCR of cycle duration div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*cycle_duration_ext*: This 9-bit field indicates the duration in NCR of cycle duration modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*dwell_duration_base*: This 33-bit field indicates the duration in NCR of dwell duration div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*dwell_duration_ext*: This 9-bit field indicates the duration in NCR of dwell duration modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*on_time_base*: This 33-bit field indicates the NCR of on_time div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*on_time_ext*: This 9-bit field indicates the NCR of on_time modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*bit_map_size*: This 15-bit field indicates the size in bits of the bit map containing the slot_transmission_on.

*current_slot*: This 15-bit field indicates the slot in which the transmission of this table started. When it is set to 32 767 (all bits set to 1), then the current slot is not communicated for this entry.

*slot_transmission_on*: The grid size can be calculated by dividing the cycle duration by the *bit_map_size*. Starting from the beginning of the cycle there are then *bit_map_size* timeslots of length grid_size each which are matched with the *slot_transmission_on* bits in this loop starting from the first. The *slot_transmission_on* bit indicates if there is a transmission in the respective timeslot.

*padding_bit*: Padding bits are added after the *slot_transmission_on* bit map to reach again alignment with 8 bits. The number J of padding bits is equal to 7 - ((*bit_map_size* - 1) mod 8) and falls thus in the range from 0 to 7. Set value to 0.

    EXAMPLE:     If *bit_map_size* is 513, then the number of padding bits is 7. The sum of 513+7 bits then becomes 520 bits which is an integer multiple of 8 bit and gives the 8 bit alignment.

*grid_size_base*: This 33-bit field indicates the duration in NCR of grid_size div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1]. Beamhopping can only take place at an integer multiple of this interval with a duration *grid_size*.

*grid_size_ext*: This 9-bit field indicates the duration in NCR of grid_size modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*revisit_duration_base*: This 33-bit field indicates the duration in NCR of revisit_duration div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1]. revisit_duration is the maximal duration of time by which a cell is not illuminated, when not in sleep mode. When *revisit_duration_base* and *revisit_duration_ext* are 0, then there is no such maximum duration.

*revisit_duration_ext*: This 9-bit field indicates the duration in NCR of revisit_duration modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*sleep_time_base*: This 33-bit field indicates the NCR of sleep_time div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1]. sleep_time is the NCR time when the sleep mode will be entered. Starting from this NCR time for the duration given by sleep_duration, receivers can expect not to be illuminated.

*sleep_time_ext***:** This 9-bit field indicates the NCR of sleep_time modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

*sleep_duration_base***:** This 33-bit field indicates the duration in NCR of sleep_duration div 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1]. When *sleep_duration_base* and *sleep_duration_ext* are 0, there is no such sleep mode.

*sleep_duration_ext***:** This 9-bit field indicates the duration in NCR of sleep_duration modulo 300, as specified in ETSI EN 301 790 [6] and ISO/IEC 13818-1 [1].

## 5.2.11.6     Satellite Position v3 info

The `satellite_position_v3_info` provides an alternative way of specifying the satellite ephemeris data. This method uses state vectors rather than the orbit averages underlying the format used in `satellite_position_v2_info`. It is further based on Section 5 of [i.11]. Because the format is intended to contain data for members of a particular satellite network, certain parameters defined in [i.11] are assumed common to all satellites and known to the user of the data by external means.

The format is defined in Table 11h. When the data set for one satellite needs to be split between several sections, the "Metadata", as indicated by the metadata_flag, should be included only in the first section. Further, co-variance data, if used, should be included only once per data set.

**Table 11h: Satellite position v3 info**

| Syntax | Number of Bits | Mnemonic |
|---|---|---|
| satellite_position_v3_info () { | | |
| oem_version_major | 4 | uimsbf |
| oem_version_minor | 4 | uimsbf |
| creation_date_year | 8 | uimsbf |
| reserved_zero_future_use | 7 | bslbf |
| creation_date_day | 9 | uimsbf |
| creation_date_day_fraction | 32 | spfmsbf |
| for (i=1; i<=N; i++) { | | |
| satellite_id | 24 | bslbf |
| reserved_zero_future_use | 3 | bslbf |
| metadata_flag | 1 | bslbf |
| usable_start_time_flag | 1 | bslbf |
| usable_stop_time_flag | 1 | bslbf |
| ephemeris_accel_flag | 1 | bslbf |
| covariance_flag | 1 | bslbf |
| if (metadata_flag == 1) { | | |
| total_start_time_year | 8 | uimsbf |
| reserved_zero_future_use | 7 | bslbf |
| total_start_time_day | 9 | uimsbf |
| total_start_time_day_fraction | 32 | spfmsbf |
| total_stop_time_year | 8 | uimsbf |
| reserved_zero_future_use | 7 | bslbf |
| total_stop_time_day | 9 | uimsbf |
| total_stop_time_day_fraction | 32 | spfmsbf |
| reserved_zero_future_use | 1 | bslbf |
| interpolation_flag | 1 | bslbf |
| interpolation_type | 3 | uimsbf |
| interpolation_degree | 3 | uimsbf |
| if (usable_start_time_flag == 1) { | | |
| usable_start_time_year | 8 | uimsbf |
| reserved_zero_future_use | 7 | bslbf |
| usable_start_time_day | 9 | uimsbf |
| usable_start_time_day_fraction | 32 | spfmsbf |
| } | | |
| if (usable_stop_time_flag == 1) { | | |
| usable_stop_time_year | 8 | uimsbf |
| reserved_zero_future_use | 7 | bslbf |
| usable_stop_time_day | 9 | uimsbf |

| Syntax | Number of Bits | Mnemonic |
|---|---|---|
| ` usable_stop_time_day_fraction` | 32 | spfmsbf |
| `  }` | | |
| `}` | | |
| `ephemeris_data_count` | 16 | uimsbf |
| `for (j=0; j< ephemeris_data_count; j++) {` | | |
| `epoch_year` | 8 | uimsbf |
| `reserved_zero_future_use` | 7 | bslbf |
| `epoch_day` | 9 | uimsbf |
| `epoch_day_fraction` | 32 | spfmsbf |
| `ephemeris_x` | 32 | spfmsbf |
| `ephemeris_y` | 32 | spfmsbf |
| `ephemeris_z` | 32 | spfmsbf |
| `ephemeris_x_dot` | 32 | spfmsbf |
| `ephemeris_y_dot` | 32 | spfmsbf |
| `ephemeris_z_dot` | 32 | spfmsbf |
| `if (ephemeris_accel_flag) {` | | |
| `ephemeris_x_ddot` | 32 | spfmsbf |
| `ephemeris_y_ddot` | 32 | spfmsbf |
| `ephemeris_z_ddot` | 32 | spfmsbf |
| `}` | | |
| `}` | | |
| `if (covariance_flag == 1) {` | | |
| `covariance_epoch_year` | 8 | uimsbf |
| `reserved_zero_future_use` | 7 | bslbf |
| `covariance_epoch_day` | 9 | uimsbf |
| `covariance_epoch_day_fraction` | 32 | spfmsbf |
| `for (j=0; j<21; j++) {` | | |
| `covariance_element` | 32 | spfmsbf |
| `}` | | |
| `}` | | |
| `}` | | |
| `}` | | |

Semantics for satellite_position_v3_info:

- **oem_version_major**, **oem_version_minor:** These 4-bit field version indicators respectively contain the major and minor version numbers of the OEM standard underlying the data record.

- **creation_date_year:** This 8-bit field contains the last 2 digits of the year of the creation of the data set. Valid values are 0 to 99 inclusive.

- **creation_date_day:** This 9-bit field contains the day of the year of the creation of the data set. Valid values are 1 to 366.

- **creation_date_day_fraction:** This 32-bit field contains the fraction of the day of the creation of the data set. Valid values are 0,0 to1,0.

- **satellite_id:** This is a 24-bit field which serves as a label to identify the satellite that is detailed in this loop.

- **metadata_flag:** This 1-bit flag, when set to "1", indicates that the data set for this satellite includes the metadata group. This flag should be set to '1' in the first segment containing data for a given satellite, and to '0' in the remaining segments.

- **usable_start_time_flag:** This 1-bit flag, when set to "1", indicates that the data set for this satellite includes the optional start time for the usable span covered by the ephemeris data. If metadata_flag == 0, this value is reserved.

- **usable_stop_time_flag:** This 1-bit flag, when set to "1", indicates that the data set for this satellite includes the optional stop time for the usable span covered by the ephemeris data. If metadata_flag == 0, this value is reserved.

- **ephemeris_accel_flag:** This 1-bit flag, when set to "1", indicates that all ephemeris data records for this satellite include the optional acceleration elements ephemeris_x_ddot, ephemeris_y_ddot and ephemeris_z_ddot.

- **covariance_flag:** This 1-bit flag, when set to "1", indicates that the data set for this satellite includes the optional covariance matrix in the current section.

- **total_start_time_year:** This 8-bit field contains the last 2 digits of the year of the start of the total time span covered by the ephemeris data and optional covariance data. Valid values are 0 to 99 inclusive.

- **total_start_time_day:** This 9-bit field contains the day of the year of the start of the total time span covered by the ephemeris data and optional covariance data. Valid values are 1 to 366.

- **total_start_time_day_fraction:** This 32-bit field contains the fraction of the day of the year of the start of the total time span covered by the ephemeris data and optional covariance data. Valid values are 0,0 to 1,0.

- **total_stop_time_year:** This 8-bit field contains the last 2 digits of the year of the end of the total time span covered by the ephemeris data and optional covariance data. Valid values are 0 to 99 inclusive.

- **total_stop_time_day:** This 9-bit field contains the day of the year of the end of the total time span covered by the ephemeris data and optional covariance data. Valid values are 1 to 366.

- **total_stop_time_day_fraction:** This 32-bit field contains the fraction of the day of the year of the end of the total time span covered by the ephemeris data and optional covariance data. Valid values are 0,0 to 1.0.

- **interpolation_flag:** This 1-bit flag, when set to "1", indicates that the data set for this satellite includes the optional definition of an interpolation method.

- **interpolation_type:** This 3-bit field indicates the recommended interpolation method, as defied in Table 11i. If interpolation_flag == 0, this field is reserved.

**Table 11i: Interpolation method for ephemeris data**

| Value | Method |
|---|---|
| 0 | Reserved |
| 1 | Linear |
| 2 | Lagrange |
| 3 | Reserved |
| 4 | Hermite |
| 5 to 7 | Reserved |

- **interpolation_degree:** This 3-bit field indicates the recommended order of the interpolation. If interpolation_flag == 0, this field is reserved.

- **usable_start_time_year:** This 8-bit field contains the last 2 digits of the year of the start of the time span covered by the ephemeris data. Valid values are 0 to 99 inclusive.

- **usable_start_time_day:** This 9-bit field contains the day of the year of the start of the time span covered by the ephemeris data. Valid values are 1 to 366.

- **usable_start_time_day_fraction:** This 32-bit field contains the fraction of the day of the year of the start of the time span covered by the ephemeris data. Valid values are 0,0 to 1,0.

- **usable_stop_time_year:** This 8-bit field contains the last 2 digits of the year of the end of the time span covered by the ephemeris data. Valid values are 0 to 99 inclusive.

- **usable_stop_time_day:** This 9-bit field contains the day of the year of the end of the time span covered by the ephemeris data. Valid values are 1 to 366.

- **usable_stop_time_day_fraction:** This 32-bit field contains the fraction of the day of the year of the end of the time span covered by the ephemeris data. Valid values are 0,0 to 1,0.

- **ephemeris_data_count:** This 16-bit field contains the number of ephemeris data records that follow.

- **epoch_year:** This 8-bit field contains the last 2 digits of the year of the ephemeris data record. Valid values are 0 to 99 inclusive.

- **epoch_day:** This 9-bit field contains the day of the year of the ephemeris data record. Valid values are 1 to 366.

- **epoch_day_fraction:** This 32-bit field contains the fraction of the day of the year of the start of the t ephemeris data record. Valid values are 0,0 to 1,0.

- **ephemeris_x**, **ephemeris_y**, **ephemeris_z:** These 32-bit fields contain the cartesian coordinates of the satellite, in meters.

- **ephemeris_x_dot**, e**phemeris_y_dot**, **ephemeris_z_dot:** These 32-bit fields contain the velocity terms in the directions of the axes of the cartesian coordinate system, in meters per second.

- **ephemeris_x_ddot**, **ephemeris_y_ddot**, **ephemeris_z_ddot:** These 32-bit fields contain the acceleration terms in the directions of the axes of the cartesian coordinate system, in meters per second per second.

- **covariance_epoch_year:** This 8-bit field contains the last 2 digits of the year of the epoch of the covariance matrix. Valid values are 0 to 99 inclusive.

- **covariance_epoch_day:** This 9-bit field contains the day of the year of the epoch of the covariance matrix. Valid values are 1 to 366.

- **covariance_epoch_day_fraction:** This 32-bit field contains the fraction of the day of the year of the epoch of the covariance matrix. Valid values are 0,0 to 1,0.

- **covariance_element:** This 32-bit field contains one element of the covariance matrix. The elements shall be ordered sequentially from upper left [1,1] to lower right [6,6], lower triangular form, row by row left to right.

# 6  Descriptors

## 6.0  Introduction

This clause describes the different descriptors that can be used within the SI (for further information refer to ETSI TS 101 211 [i.1]).

## 6.1  Descriptor identification and location

Table 12 lists the descriptors declared or defined within the present document, giving the descriptors-tag values and the intended placement within the SI tables. This does not imply that their use in other tables is restricted.

Table 12 uses the following mark-up to indicate the possible locations of descriptors:

- a check mark ("✓ ") indicates that the descriptor may be carried in the respective table;

- a dash ("-") indicates that the descriptor shall not be carried in the respective table;

- an empty table cell indicates that nothing is implied regarding the carriage of the descriptor in the respective table.

**Table 12: Possible locations of descriptors**

| Descriptor | Tag value | NIT | BAT | SDT | EIT | TOT | PMT | SIT (see note 1) |
|---|---|---|---|---|---|---|---|---|
| network_name_descriptor | 0x40 | ✓ | - | - | - | - | - | - |
| service_list_descriptor | 0x41 | ✓ | ✓ | - | - | - | - | - |
| stuffing_descriptor | 0x42 | ✓ | ✓ | ✓ | ✓ | - | - | ✓ |
| satellite_delivery_system_descriptor | 0x43 | ✓ | - | - | - | - | - | - |
| cable_delivery_system_descriptor | 0x44 | ✓ | - | - | - | - | - | - |

| Descriptor | Tag value | NIT | BAT | SDT | EIT | TOT | PMT | SIT (see note 1) |
|---|---|---|---|---|---|---|---|---|
| VBI_data_descriptor | 0x45 | - | - | - | - | - | ✓ | - |
| VBI_teletext_descriptor | 0x46 | - | - | - | - | - | ✓ | - |
| bouquet_name_descriptor | 0x47 | - | ✓ | - | - | - | - | ✓ |
| service_descriptor | 0x48 | - | - | ✓ | - | - | - | ✓ |
| country_availability_descriptor | 0x49 | - | ✓ | ✓ | - | - | - | ✓ |
| linkage_descriptor | 0x4A | ✓ | ✓ | ✓ | ✓ | - | - | ✓ |
| NVOD_reference_descriptor | 0x4B | - | - | ✓ | - | - | - | ✓ |
| time_shifted_service_descriptor | 0x4C | - | - | ✓ | - | - | - | ✓ |
| short_event_descriptor | 0x4D | - | - | - | ✓ | - | - | ✓ |
| extended_event_descriptor | 0x4E | - | - | - | ✓ | - | - | ✓ |
| time_shifted_event_descriptor | 0x4F | - | - | - | ✓ | - | - | ✓ |
| component_descriptor | 0x50 | - | - | ✓ | ✓ | - | - | ✓ |
| mosaic_descriptor | 0x51 | - | - | ✓ | - | - | ✓ | ✓ |
| stream_identifier_descriptor | 0x52 | - | - | - | - | - | ✓ | - |
| CA_identifier_descriptor | 0x53 | - | ✓ | ✓ | ✓ | - | - | ✓ |
| content_descriptor | 0x54 | - | - | - | ✓ | - | - | ✓ |
| parental_rating_descriptor | 0x55 | - | - | - | ✓ | - | - | ✓ |
| teletext_descriptor | 0x56 | - | - | - | - | - | ✓ | - |
| telephone_descriptor | 0x57 | - | - | ✓ | ✓ | - | - | ✓ |
| local_time_offset_descriptor | 0x58 | - | - | - | - | ✓ | - | - |
| subtitling_descriptor | 0x59 | - | - | - | - | - | ✓ | - |
| terrestrial_delivery_system_descriptor | 0x5A | ✓ | - | - | - | - | - | - |
| multilingual_network_name_descriptor | 0x5B | ✓ | - | - | - | - | - | - |
| multilingual_bouquet_name_descriptor | 0x5C | - | ✓ | - | - | - | - | - |
| multilingual_service_name_descriptor | 0x5D | - | - | ✓ | - | - | - | ✓ |
| multilingual_component_descriptor | 0x5E | - | - | - | ✓ | - | - | ✓ |
| private_data_specifier_descriptor | 0x5F | ✓ | ✓ | ✓ | ✓ | - | ✓ | ✓ |
| service_move_descriptor | 0x60 | - | - | - | - | - | ✓ | - |
| short_smoothing_buffer_descriptor | 0x61 | - | - | - | ✓ | - | - | ✓ |
| frequency_list_descriptor | 0x62 | ✓ | - | - | - | - | - | - |
| partial_transport_stream_descriptor | 0x63 | - | - | - | - | - | - | ✓ |
| data_broadcast_descriptor | 0x64 | - | - | ✓ | ✓ | - | - | ✓ |
| scrambling_descriptor | 0x65 | - | - | - | - | - | ✓ | - |
| data_broadcast_id_descriptor | 0x66 | - | - | - | - | - | ✓ | - |
| transport_stream_descriptor (see note 2) | 0x67 | - | - | - | - | - | - | - |
| DSNG_descriptor (see note 2) | 0x68 | - | - | - | - | - | - | - |
| PDC_descriptor | 0x69 | - | - | - | ✓ | - | - | - |
| AC-3_descriptor (see annex D) | 0x6A | - | - | - | - | - | ✓ | - |
| ancillary_data_descriptor | 0x6B | - | - | - | - | - | ✓ | - |
| cell_list_descriptor | 0x6C | ✓ | - | - | - | - | - | - |
| cell_frequency_link_descriptor | 0x6D | ✓ | - | - | - | - | - | - |
| announcement_support_descriptor | 0x6E | - | - | ✓ | - | - | - | - |
| application_signalling_descriptor (see ETSI TS 102 809 [25]) | 0x6F | - | - | - | - | - | ✓ | - |
| adaptation_field_data_descriptor | 0x70 | - | - | - | - | - | ✓ | - |
| service_identifier_descriptor (see ETSI TS 102 812 [26]) | 0x71 | - | - | ✓ | - | - | - | - |
| service_availability_descriptor | 0x72 | - | - | ✓ | - | - | - | - |
| default_authority_descriptor (see ETSI TS 102 323 [21]) | 0x73 | ✓ | ✓ | ✓ | - | - | - | - |
| related_content_descriptor (see ETSI TS 102 323 [21]) | 0x74 | - | - | - | - | - | ✓ | - |
| TVA_id_descriptor (see ETSI TS 102 323 [21]) | 0x75 | - | - | - | ✓ | - | - | - |
| content_identifier_descriptor (see ETSI TS 102 323 [21]) | 0x76 | - | - | - | ✓ | - | - | - |
| time_slice_fec_identifier_descriptor (see note 3) | 0x77 | ✓ | - | - | - | - | - | - |
| ECM_repetition_rate_descriptor (ETSI EN 301 192 [3]) | 0x78 | - | - | - | - | - | ✓ | - |
| S2_satellite_delivery_system_descriptor | 0x79 | ✓ | - | - | - | - | - | - |
| enhanced_AC-3_descriptor (see annex D) | 0x7A | - | - | - | - | - | ✓ | - |
| DTS_descriptor (see annex G) | 0x7B | - | - | - | - | - | ✓ | - |

| Descriptor | Tag value | NIT | BAT | SDT | EIT | TOT | PMT | SIT (see note 1) |
|---|---|---|---|---|---|---|---|---|
| `AAC_descriptor` (see annex H) | 0x7C | - | - | - | - | - | ✓ | - |
| `XAIT_location_descriptor` (ETSI TS 102 727 [i.2]) | 0x7D | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| `FTA_content_management_descriptor` | 0x7E | ✓ | ✓ | ✓ | ✓ | - | - | - |
| `extension_descriptor` (see note 4) | 0x7F | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| user defined | 0x80 to 0xFE | | | | | | | |
| reserved for future use | 0xFF | | | | | | | |

NOTE 1:   Only found in Partial Transport Streams.
NOTE 2:   Only in the TSDT.
NOTE 3:   May also be located in the CAT (ISO/IEC 13818-1 [1]) and IP/MAC Notification Table (INT) (ETSI EN 301 192 [3]).
NOTE 4:   See also clause 6.3 and clause 6.4.

# 6.2    Descriptor coding

## 6.2.0    General principles

When the construct descriptor() appears in the sections of clause 5.2, this indicates that zero or more of the descriptors defined in clause 6.2 shall occur.

The following semantics apply to all the descriptors defined in clause 6.2.

*descriptor_tag***:** The *descriptor_tag* is an 8-bit field which identifies each descriptor. Those values with MPEG-2 normative meaning are described in ISO/IEC 13818-1 [1]. The values of *descriptor_tag* are defined in table 12.

*descriptor_length***:** The *descriptor_length* is an 8-bit field specifying the total number of bytes of the data portion of the descriptor following the byte defining the value of this field.

The bit and transmission ordering rules defined in clause 5.1.6 shall apply.

## 6.2.1    Adaptation field data descriptor

The adaptation_field_data_descriptor (see table 13) provides a means of indicating the type of data fields supported within the private data field of the adaptation field. These data fields shall be coded according to annex D of ETSI TS 101 154 [14]. This descriptor shall be inserted into the corresponding *ES_info* loop of the PMT if the stream contains one or more of the data fields listed in table 14.

**Table 13: Adaptation field data descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| adaptation_field_data_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    adaptation_field_data_identifier | 8 | bslbf |
| } | | |

Semantics for the adaptation field data descriptor:

*adaptation_field_data_identifier***:** This is an 8-bit field identifying data fields transmitted in the private data bytes of the adaptation field. It shall be coded according to table 14. If a bit in the *adaptation_field_data_identifier* is set to 0b1 it indicates that the transmission of the corresponding data field (as specified in the standard identified in the description column) is supported.

NOTE:    The data field does not necessarily occur in every adaptation field.

**Table 14: Adaptation field data identifier coding**

| adaptation_field_data_identifier bit | Description |
|---|---|
| $b_0$ (see note) | announcement_switching_data |
| $b_1$ | AU_information |
| $b_2$ | PVR_assist_information |
| $b_3$ | reserved_zero_future_use |
| $b_4$ | reserved_zero_future_use |
| $b_5$ | reserved_zero_future_use |
| $b_6$ | reserved_zero_future_use |
| $b_7$ | reserved_zero_future_use |
| NOTE: This bit is transmitted last (see clause 5.1.6). | |

## 6.2.2 Ancillary data descriptor

The `ancillary_data_descriptor` (see table 15) provides a means of indicating the presence and the type of ancillary data in audio elementary streams coded according to clause 6.1 of ETSI TS 101 154 [14]. It shall be inserted into the corresponding *ES_info* loop of the PMT. If the ancillary data adheres to one of the formats in table 16, the descriptor shall be present.

**Table 15: Ancillary data descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| ancillary_data_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    ancillary_data_identifier | 8 | bslbf |
| } | | |

Semantics for the ancillary data descriptor:

*ancillary_data_identifier*: This is an 8-bit field identifying ancillary data coded in the audio elementary stream. It shall be coded according to table 16. If a bit in the *ancillary_data_identifier* field is set to 0b1 it indicates that ancillary data includes the corresponding data field.

**Table 16: Ancillary data identifier coding**

| ancillary_data_identifier bit | Description |
|---|---|
| $b_0$ (see note) | Digital Versatile Disc (DVD) Video Ancillary Data (ETSI TS 101 154 [14]) |
| $b_1$ | Extended Ancillary Data (ETSI TS 101 154 [14]) |
| $b_2$ | Announcement Switching Data (ETSI TS 101 154 [14]) |
| $b_3$ | Digital Audio Broadcasting (DAB) Ancillary Data (ETSI EN 300 401 [29]) |
| $b_4$ | Scale Factor Error Check (ScF-CRC) (ETSI TS 101 154 [14]) |
| $b_5$ | MPEG-4 ancillary data (ETSI TS 101 154 [14], clause C.5) |
| $b_6$ | Radio Data System (RDS) via Universal Encoder Communication Protocol (UECP) (ETSI TS 101 154 [14]) |
| $b_7$ | reserved_zero_future_use |
| NOTE: This bit is transmitted last (see clause 5.1.6). | |

## 6.2.3 Announcement support descriptor

The `announcement_support_descriptor` (see table 17) identifies the type of announcements that are supported by the service. Furthermore, it informs about the transport method of the announcement and gives the necessary linkage information so that the announcement stream can be monitored.

**Table 17: Announcement support descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| announcement_support_descriptor() { | | |
|   descriptor_tag | 8 | uimsbf |
|   descriptor_length | 8 | uimsbf |
|   announcement_support_indicator | 16 | bslbf |
|   for (i=0;i<N;i++) { | | |
|     announcement_type | 4 | uimsbf |
|     reserved_future_use | 1 | bslbf |
|     reference_type | 3 | uimsbf |
|     if (reference_type == 0x01 | | |
|     \|\| reference_type == 0x02 | | |
|     \|\| reference_type == 0x03) { | | |
|       original_network_id | 16 | uimsbf |
|       transport_stream_id | 16 | uimsbf |
|       service_id | 16 | uimsbf |
|       component_tag | 8 | uimsbf |
|     } | | |
|   } | | |
| } | | |

Semantics for the announcement support descriptor:

*announcement_support_indicator*: The announcement support indicator is a 16-bit flag field specifying which types of announcements are supported by the service. The field shall be coded according to clause C.4.3 of ETSI TS 101 154 [14].

**Table 18: Void**

*announcement_type*: This 4-bit field specifies the type of announcement for which the following fields in the loop are valid, see table 19 coding.

**Table 19: Announcement type coding**

| announcement_type | Description |
|---|---|
| 0b0000 | Emergency alarm |
| 0b0001 | Road Traffic flash |
| 0b0010 | Public Transport flash |
| 0b0011 | Warning message |
| 0b0100 | News flash |
| 0b0101 | Weather flash |
| 0b0110 | Event announcement |
| 0b0111 | Personal call |
| 0b1000 to 0b1111 | Reserved for future use |

*reference_type*: This is a 3-bit field. It specifies the transport method of the announcement according to table 20.

**Table 20: Reference type coding**

| reference_type | Description |
|---|---|
| 0b000 | Announcement is broadcast in the usual audio stream of the service |
| 0b001 | Announcement is broadcast in a separate audio stream that is part of the service |
| 0b010 | Announcement is broadcast by means of a different service within the same DVB transport stream |
| 0b011 | Announcement is broadcast by means of a different service within a different DVB transport stream |
| 0b100 to 0b111 | Reserved for future use |

*original_network_id*: This 16-bit field gives the label identifying the *network_id* of the originating delivery system of the announcement service indicated.

*Transport_stream_id*: This is a 16-bit field which uniquely identifies the TS containing the announcement service indicated.

*Service_id***:** This is a 16-bit field which uniquely identifies the service containing the announcements indicated.

*Component_tag***:** This 8-bit field has the same value as the *component_tag* field in the stream identifier descriptor that shall be present in the PSI program map section for the audio stream on which the announcement is broadcast.

## 6.2.4    Bouquet name descriptor

The `bouquet_name_descriptor` provides the bouquet name in text form, see table 21.

**Table 21: Bouquet name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| bouquet_name_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the bouquet name descriptor:

*char***:** This is an 8-bit field, a sequence of which conveys the name of the bouquet about which the BAT sub_table informs. Text information is coded using the character sets and methods described in annex A.

## 6.2.5    CA identifier descriptor

The `CA_identifier_descriptor` (see table 22) indicates whether a particular bouquet, service, or event is associated with a conditional access system and identifies the CA system type by means of the *CA_system_id*.

**Table 22: CA identifier descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| CA_identifier_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       CA_system_id | 16 | uimsbf |
|    } | | |
| } | | |

Semantics for the CA identifier descriptor:

*CA_system_id***:** This 16-bit field identifies the CA system. It shall be coded according to ETSI TS 101 162 [15].

## 6.2.6    Cell frequency link descriptor

The `cell_frequency_link_descriptor` (see table 23) may be used in the NIT that describes a terrestrial network. It gives a complete list of cells and identifies the frequencies that are in use in these cells for the multiplex described.

**Table 23: Cell frequency link descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| cell_frequency_link_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       cell_id | 16 | uimsbf |
|       frequency | 32 | uimsbf |
|       subcell_info_loop_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          cell_id_extension | 8 | uimsbf |
|          transposer_frequency | 32 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the cell frequency link descriptor:

*cell_id*: This is a 16-bit field which uniquely identifies a cell in the terrestrial network.

*frequency*: This 32-bit field identifies the main frequency that is used in the cell indicated. The coding is according to the coding of the *centre_frequency* in the terrestrial_delivery_system_descriptor.

*subcell_info_loop_length*: This 8-bit field conveys the total length in bytes of the following loop that describes the subcells.

*cell_id_extension*: This 8-bit field is used to identify a subcell within a cell.

*transposer_frequency*: This 32-bit field indicates the centre frequency that is used by a transposer in the sub-cell indicated. It is encoded in the same way as the *centre_frequency* field.

## 6.2.7    Cell list descriptor

The cell_list_descriptor (see table 24) may be used in the NIT that describes a terrestrial network. It provides a list of all cells of the network about which the NIT sub_table informs and describes their coverage areas.

**Table 24: Cell list descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| cell_list_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       cell_id | 16 | uimsbf |
|       cell_latitude | 16 | uimsbf |
|       cell_longitude | 16 | uimsbf |
|       cell_extent_of_latitude | 12 | uimsbf |
|       cell_extent_of_longitude | 12 | uimsbf |
|       subcell_info_loop_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          cell_id_extension | 8 | uimsbf |
|          subcell_latitude | 16 | uimsbf |
|          subcell_longitude | 16 | uimsbf |
|          subcell_extent_of_latitude | 12 | uimsbf |
|          subcell_extent_of_longitude | 12 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the cell list descriptor:

*cell_id***:** This is a 16-bit field which uniquely identifies a cell in the terrestrial network.

*cell_latitude***:** This 16-bit field, coded as a two's complement number, shall specify the latitude of the south-western corner of a spherical rectangle that approximately describes the coverage area of the cell indicated. It shall be calculated by multiplying the value of the latitude field by $\frac{90°}{2^{15}}$. Southern latitudes shall be considered negative and northern latitudes positive.

*cell_longitude***:** This 16-bit field, coded as a two's complement number, shall specify the longitude of the south-western corner of a spherical rectangle that approximately describes the coverage area of the cell indicated. It shall be calculated by multiplying the value of the longitude field by $\frac{180°}{2^{15}}$. Western longitudes shall be considered negative and eastern longitudes positive.

*cell_extent_of_latitude***:** This 12-bit field, coded as an unsigned binary number, shall specify the extent of latitude of a spherical rectangle that approximately describes the coverage area of the cell indicated. It shall be calculated by multiplying the value of the extent of latitude field by $\frac{90°}{2^{15}}$.

*cell_extent_of_longitude***:** This 12-bit field, coded as an unsigned binary number, shall specify the extent of longitude of a spherical rectangle that approximately describes the coverage area of the cell indicated. It shall be calculated by multiplying the value of the extent of longitude field by $\frac{180°}{2^{15}}$.

*subcell_info_loop_length***:** This 8-bit field conveys the total length in bytes of the following loop that describes the subcells.

*cell_id_extension***:** This 8-bit field is used to identify a subcell within a cell.

*subcell_latitude***:** This 16-bit field, coded as a two's complement number, shall specify the latitude of the south-western corner of a spherical rectangle that approximately describes the coverage area of the subcell indicated. It shall be calculated by multiplying the value of the latitude field by $\frac{90°}{2^{15}}$. Southern latitudes shall be considered negative and northern latitudes positive.

*subcell_longitude***:** This 16-bit field, coded as a two's complement number, shall specify the longitude of the south-western corner of a spherical rectangle that approximately describes the coverage area of the subcell indicated. It shall be calculated by multiplying the value of the longitude field by $\frac{180°}{2^{15}}$. Western longitudes shall be considered negative and eastern longitudes positive.

*subcell_extent_of_latitude***:** This 12-bit field, coded as an unsigned binary number, shall specify the extent of latitude of a spherical rectangle that approximately describes the coverage area of the subcell indicated. It shall be calculated by multiplying the value of the extent of latitude field by $\frac{90°}{2^{15}}$.

*subcell_extent_of_longitude***:** This 12-bit field, coded as an unsigned binary number, shall specify the extent of longitude of a spherical rectangle that approximately describes the coverage area of the subcell indicated. It shall be calculated by multiplying the value of the extent of longitude field by $\frac{180°}{2^{15}}$.

## 6.2.8    Component descriptor

The component_descriptor identifies the type of component stream and may be used to provide a text description of the elementary stream (see table 25). When used in the SDT and EIT these component stream characteristics, as are defined in table 26, describe the editorial characteristics of the event or service. These are intended to be used by receivers to inform the user of these characteristics. The profiles and levels of the codecs mentioned in table 26 are as defined in ETSI TS 101 154 [14] and ETSI TS 102 005 [19].

Note that the use of some characteristics relating to High Efficiency Video Coding (HEVC) video components (*stream_content* 0x9 and *stream_content_ext* 0x0) listed in table 26 imply additional semantics and are subject to additional rules (also see note 9 in table 26).

**Table 25: Component descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| component_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    stream_content_ext | 4 | uimsbf |
|    stream_content | 4 | uimsbf |
|    component_type | 8 | uimsbf |
|    component_tag | 8 | uimsbf |
|    ISO_639_language_code | 24 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the component descriptor:

*stream_content_ext***:** This 4-bit field in combination with the *stream_content* field specifies the type of stream. The coding of this field is specified in table 26. If the value of the *stream_content* field is in the range 0x1 to 0x8, the *stream_content_ext* field shall be set to 0xF to maintain backward compatibility with previous versions of the present document.

*stream_content***:** This 4-bit field in combination with the *stream_content_ext* field specifies the type of stream. The coding of this field is specified in table 26.

*component_type***:** This 8-bit field specifies the type of the component. The coding of this field is specified in table 26.

*component_tag***:** This 8-bit field has the same value as the *component_tag* field in the stream_identifier_descriptor (if present in the PSI program_map_section) for the component stream. Multiple component descriptors for the same *component_tag* value are permitted as defined in table 26.

*ISO_639_language_code***:** This 24-bit field identifies the language of the component (in the case of audio or EBU-data) and of the text description which may be contained in this descriptor. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

    EXAMPLE:    French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101

*char***:** This is an 8-bit field, a sequence of which conveys a text description of the component stream. Text information is coded using the character sets and methods described in annex A.

**Table 26: stream_content, stream_content_ext, and component_type coding**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0x0 | 0x0 to 0xF | 0x00 to 0xFF | reserved for future use |
| 0x1 | n/a (see note 1) | 0x00 | reserved for future use |
| | | 0x01 | MPEG-2 video, 4:3 aspect ratio, 25 Hz (see note 2) |
| | | 0x02 | MPEG-2 video, 16:9 aspect ratio with pan vectors, 25 Hz (see note 2) |
| | | 0x03 | MPEG-2 video, 16:9 aspect ratio without pan vectors, 25 Hz (see note 2) |
| | | 0x04 | MPEG-2 video, > 16:9 aspect ratio, 25 Hz (see note 2) |
| | | 0x05 | MPEG-2 video, 4:3 aspect ratio, 30 Hz (see note 2) |
| | | 0x06 | MPEG-2 video, 16:9 aspect ratio with pan vectors, 30 Hz (see note 2) |
| | | 0x07 | MPEG-2 video, 16:9 aspect ratio without pan vectors, 30 Hz (see note 2) |
| | | 0x08 | MPEG-2 video, > 16:9 aspect ratio, 30 Hz (see note 2) |
| | | 0x09 | MPEG-2 high definition video, 4:3 aspect ratio, 25 Hz (see note 2) |
| | | 0x0A | MPEG-2 high definition video, 16:9 aspect ratio with pan vectors, 25 Hz (see note 2) |

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| | | 0x0B | MPEG-2 high definition video, 16:9 aspect ratio without pan vectors, 25 Hz (see note 2) |
| | | 0x0C | MPEG-2 high definition video, > 16:9 aspect ratio, 25 Hz (see note 2) |
| | | 0x0D | MPEG-2 high definition video, 4:3 aspect ratio, 30 Hz (see note 2) |
| | | 0x0E | MPEG-2 high definition video, 16:9 aspect ratio with pan vectors, 30 Hz (see note 2) |
| | | 0x0F | MPEG-2 high definition video, 16:9 aspect ratio without pan vectors, 30 Hz (see note 2) |
| | | 0x10 | MPEG-2 high definition video, > 16:9 aspect ratio, 30 Hz (see note 2) |
| | | 0x11 to 0xAF | reserved for future use |
| | | 0xB0 to 0xFE | user defined |
| | | 0xFF | reserved for future use |
| 0x2 | n/a (see note 1) | 0x00 | reserved for future use |
| | | 0x01 | MPEG-1 Layer 2 audio, single mono channel |
| | | 0x02 | MPEG-1 Layer 2 audio, dual mono channel |
| | | 0x03 | MPEG-1 Layer 2 audio, stereo (2 channel) |
| | | 0x04 | MPEG-1 Layer 2 audio, multi-lingual, multi-channel |
| | | 0x05 | MPEG-1 Layer 2 audio, surround sound |
| | | 0x06 to 0x3F | reserved for future use |
| | | 0x40 | MPEG-1 Layer 2 audio description for the visually impaired (see note 3) |
| | | 0x41 | MPEG-1 Layer 2 audio for the hard of hearing |
| | | 0x42 | receiver-mix supplementary audio as per annex E of ETSI TS 101 154 [14] |
| | | 0x43 to 0x46 | reserved for future use |
| | | 0x47 | MPEG-1 Layer 2 audio, receiver-mix audio description |
| | | 0x48 | MPEG-1 Layer 2 audio, broadcast-mix audio description |
| | | 0x49 to 0xAF | reserved for future use |
| | | 0xB0 to 0xFE | user defined |
| | | 0xFF | reserved for future use |
| 0x3 | n/a (see note 1) | 0x00 | reserved for future use |
| | | 0x01 | EBU Teletext subtitles |
| | | 0x02 | associated EBU Teletext |
| | | 0x03 | Vertical Blanking Interval (VBI) data |
| | | 0x04 to 0x0F | reserved for future use |
| | | 0x10 | DVB subtitles ETSI EN 300 743 [2] (normal) with no monitor aspect ratio criticality |
| | | 0x11 | DVB subtitles ETSI EN 300 743 [2] (normal) for display on 4:3 aspect ratio monitor |
| | | 0x12 | DVB subtitles ETSI EN 300 743 [2] (normal) for display on 16:9 aspect ratio monitor |
| | | 0x13 | DVB subtitles ETSI EN 300 743 [2] (normal) for display on 2.21:1 aspect ratio monitor |
| | | 0x14 | DVB subtitles ETSI EN 300 743 [2] (normal) for display on a high definition monitor |
| | | 0x15 | DVB subtitles ETSI EN 300 743 [2] (normal) with plano-stereoscopic disparity for display on a high definition monitor |
| | | 0x16 | DVB subtitles ETSI EN 300 743 [2] (normal) for display on an ultra high definition monitor |
| | | 0x17 to 0x1F | reserved for future use |
| | | 0x20 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) with no monitor aspect ratio criticality |
| | | 0x21 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) for display on 4:3 aspect ratio monitor |
| | | 0x22 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) for display on 16:9 aspect ratio monitor |
| | | 0x23 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) for display on 2.21:1 aspect ratio monitor |
| | | 0x24 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) for display on a high definition monitor |

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| | | 0x25 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) with plano-stereoscopic disparity for display on a high definition monitor |
| | | 0x26 | DVB subtitles ETSI EN 300 743 [2] (for the hard of hearing) for display on an ultra high definition monitor |
| | | 0x27 to 0x2F | reserved for future use |
| | | 0x30 | open (in-vision) sign language interpretation for the deaf (see note 4) |
| | | 0x31 | closed sign language interpretation for the deaf (see note 4) |
| | | 0x32 to 0x3F | reserved for future use |
| | | 0x40 | video spatial resolution has been upscaled from lower resolution source material (see note 4) |
| | | 0x41 | video is Standard Dynamic Range (SDR) (see note 4) |
| | | 0x42 | video is High Dynamic Range (HDR) remapped from SDR source material (see note 4) |
| | | 0x43 | video is HDR up-converted from SDR source material (see note 4) |
| | | 0x44 | video is standard frame rate, less than or equal to 60 Hz (see note 4) |
| | | 0x45 | high frame rate video generated from lower frame rate source material (see note 4) |
| | | 0x46 to 0x7F | reserved for future use |
| | | 0x80 | dependent Spatial Audio Object Coding – Dialogue Enhancement (SAOC-DE) data stream |
| | | 0x81 to 0xAF | reserved for future use |
| | | 0xB0 to 0xFE | user defined |
| | | 0xFF | reserved for future use |
| 0x4 | n/a (see note 1) | 0x00 to 0x7F | reserved for AC-3 audio modes (refer to table D.1) |
| | | 0x80 to 0xFF | reserved for enhanced AC-3 audio modes (refer to table D.1) |
| 0x5 | n/a (see note 1) | 0x00 | reserved for future use |
| | | 0x01 | H.264/AVC standard definition video, 4:3 aspect ratio, 25 Hz (see note 2) |
| | | 0x02 | reserved for future use |
| | | 0x03 | H.264/AVC standard definition video, 16:9 aspect ratio, 25 Hz (see note 2) |
| | | 0x04 | H.264/AVC standard definition video, > 16:9 aspect ratio, 25 Hz (see note 2) |
| | | 0x05 | H.264/AVC standard definition video, 4:3 aspect ratio, 30 Hz (see note 2) |
| | | 0x06 | reserved for future use |
| | | 0x07 | H.264/AVC standard definition video, 16:9 aspect ratio, 30 Hz (see note 2) |
| | | 0x08 | H.264/AVC standard definition video, > 16:9 aspect ratio, 30 Hz (see note 2) |
| | | 0x09 to 0x0A | reserved for future use |
| | | 0x0B | H.264/AVC high definition video, 16:9 aspect ratio, 25 Hz (see note 2) |
| | | 0x0C | H.264/AVC high definition video, > 16:9 aspect ratio, 25 Hz (see note 2) |
| | | 0x0D to 0x0E | reserved for future use |
| | | 0x0F | H.264/AVC high definition video, 16:9 aspect ratio, 30 Hz (see note 2) |
| | | 0x10 | H.264/AVC high definition video, > 16:9 aspect ratio, 30 Hz (see note 2) |
| | | 0x11 to 0x7F | reserved for future use |
| | | 0x80 | H.264/AVC plano-stereoscopic frame compatible high definition video, 16:9 aspect ratio, 25 Hz, Side-by-Side (see note 2 and note 5) |
| | | 0x81 | H.264/AVC plano-stereoscopic frame compatible high definition video, 16:9 aspect ratio, 25 Hz, Top-and-Bottom (see note 2 and note 5) |

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| | | 0x82 | H.264/AVC plano-stereoscopic frame compatible high definition video, 16:9 aspect ratio, 30 Hz, Side-by-Side (see note 2, note 5, and note 6) |
| | | 0x83 | H.264/AVC stereoscopic frame compatible high definition video, 16:9 aspect ratio, 30 Hz, Top-and-Bottom (see note 2, note 5 and note 6) |
| | | 0x84 | H.264/MVC dependent view, plano-stereoscopic service compatible video (see note 5) |
| | | 0x85 to 0xAF | reserved for future use |
| | | 0xB0 to 0xFE | user defined |
| | | 0xFF | reserved for future use |
| 0x6 | n/a (see note 1) | 0x00 | reserved for future use |
| | | 0x01 | High Efficiency AAC (HE-AAC) audio, single mono channel (see note 7) |
| | | 0x02 | reserved for future use |
| | | 0x03 | HE-AAC audio, stereo (see note 7) |
| | | 0x04 | reserved for future use |
| | | 0x05 | HE-AAC audio, surround sound (see note 7) |
| | | 0x06 to 0x3F | reserved for future use |
| | | 0x40 | HE-AAC audio description for the visually impaired (see note 8 and note 7) |
| | | 0x41 | HE-AAC audio for the hard of hearing (see note 7) |
| | | 0x42 | HE-AAC receiver-mix supplementary audio as per annex E of ETSI TS 101 154 [14] (see note 7) |
| | | 0x43 | HE-AAC v2 audio, stereo |
| | | 0x44 | HE-AAC v2 audio description for the visually impaired (see note 8) |
| | | 0x45 | HE-AAC v2 audio for the hard of hearing |
| | | 0x46 | HE-AAC v2 receiver-mix supplementary audio as per annex E of ETSI TS 101 154 [14] |
| | | 0x47 | HE-AAC receiver-mix audio description for the visually impaired |
| | | 0x48 | HE-AAC broadcast-mix audio description for the visually impaired |
| | | 0x49 | HE-AAC v2 receiver-mix audio description for the visually impaired |
| | | 0x4A | HE-AAC v2 broadcast-mix audio description for the visually impaired |
| | | 0x4B to 0x9F | reserved for future use |
| | | 0xA0 | HE-AAC, or HE-AAC v2 with SAOC-DE ancillary data (see note 7 and note 4) |
| | | 0xA1 to 0xAF | reserved for future use |
| | | 0xB0 to 0xFE | user defined |
| | | 0xFF | reserved for future use |
| 0x7 | n/a (see note 1) | 0x00 to 0x7F | reserved for DTS and DTS-HD audio modes (refer to annex G) |
| | | 0x80 to 0xFF | reserved for future use |
| 0x8 | n/a (see note 1)a | 0x00 | reserved for future use |
| | | 0x01 | DVB System Renewability Message (SRM) data ETSI TS 102 770 [22] |
| | | 0x02 to 0xFF | reserved for future use for DVB Content Protection Copy Management (CPCM) modes ETSI TS 102 825 (parts 1 to 5, 7, 9 and 10) [27], ETSI TR 102 825 (parts 6, 8, 11 to 13) [i.5] |
| 0x9 | 0x0 | 0x00 | HEVC Main Profile high definition video, 50 Hz (see note 2 and note 9) |
| | | 0x01 | HEVC Main 10 Profile high definition video, 50 Hz (see note 2 and note 9) |
| | | 0x02 | HEVC Main Profile high definition video, 60 Hz (see note 2, note 6 and note 9) |
| | | 0x03 | HEVC Main 10 Profile high definition video, 60 Hz (see note 2, note 6 and note 9) |

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| | | 0x04 | HEVC ultra high definition video, with a resolution up to 3 840 x 2 160 (see note 2 and note 9, and see note 10) conformant to one of the following Ultra High Definition TeleVision (UHDTV) bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "SDR frame rate up to 60 Hz resolution up to 3 840 x 2 160";<br>• "SDR HFR dual PID with temporal scalability resolution up to 3 840 x 2 160";<br>• "HDR with HLG10 frame rate up to 60 Hz resolution up to 3 840 x 2 160";<br>• "HDR with HLG10 HFR dual PID and temporal scalability resolution up to 3 840 x 2 160". |
| | | 0x05 | HEVC ultra high definition video with Perceptual Quantizer with a bit depth of 10 bits (PQ10) HDR with a frame rate lower than or equal to 60 Hz (see note 2 and note 11) conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "HDR with PQ10 frame rate up to 60 Hz resolution up to 3 840 x 2 160"<br>or<br>HEVC ultra high definition video with PQ10 HDR HDR with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz with a half frame rate HEVC temporal video sub-bit-stream (see note 2 and note 11) conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "HDR with PQ10 HFR dual PID and temporal scalability resolution up to 3 840 x 2 160" |
| | | 0x06 | HEVC ultra high definition video, with a resolution up to 3 840 x 2 160, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream (see note 2 and note 11) conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "SDR HFR single PID resolution up to 3 840 x 2 160";<br>• "HDR with HLG10 HFR single PID resolution up to 3 840 x 2 160" |
| | | 0x07 | HEVC ultra high definition video with PQ10 HDR, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream (see note 2, and see note 11) conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "PQ10 HFR single PID resolution up to 3 840 x 2 160" |
| | | 0x08 | HEVC ultra high definition video with a resolution up to 7 680 x 4 320 (see note 2, note 6 and note 11) conformant to one of the following UHDTV2 bit stream conformance point defined in Table 18b in ETSI TS 101 154 [14]:<br>• "SDR frame rate up to 60 Hz resolution up to 7 680 x 4 320";<br>• "HDR with PQ10 frame rate up to 60 Hz resolution up to 7 680 x 4 320";<br>• "HDR with HLG10 frame rate up to 60 Hz resolution up to 7 680 x 4 320" |
| | | 0x09 to 0x0F | reserved for future use for HEVC |
| | | 0x10 | VVC Main 10 Profile with resolution up to 3 840 x 2 160 frame rate up to 60 Hz conformant to the VVC HDR UHDTV-1 bitstream conformance point defined in clause 5.15.2 in ETSI TS 101 154 [14] (see note 14) |

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| | | 0x11 | VVC Main 10 Profile with resolution up to 3 840 x 2 160 High Frame Rate of 100 Hz or 120 Hz conformant to the VVC HDR HFR UHDTV-1 bitstream conformance point defined in clause 5.15.3 in ETSI TS 101 154 [14] (see note 14) |
| | | 0x12 | VVC Main 10 Profile with resolution up to 7 680 x 4 320 frame rate up to 60 Hz  conformant to the VVC HDR UHDTV-2 bitstream conformance point defined in clause 5.15.4 in ETSI TS 101 154 [14] (see note 14) |
| | | 0x13 | VVC Main 10 Profile with resolution up to 7 680 x 4 320 High Frame Rate  of 100 Hz or 120 Hz conformant to the VVC HDR HFR UHDTV-2 bitstream conformance point defined in clause 5.15.5 in ETSI TS 101 154 [14] (see note 14) |
| | | 0x14 to 0x1F | reserved for future use for VVC |
| | | 0x20 | AVS3 High 10 Profile with resolution up to 3 840 x 2 160 frame rate up to 60 Hz conformant to the AVS3 HDR UHDTV-1 bitstream conformance point defined in clause 5.16.3 in ETSI TS 101 154 [14] (see note 15) |
| | | 0x21 | AVS3 High 10 Profile with resolution up to 3 840 x 2 160 High Frame Rate of 100 Hz or 120 Hz conformant to the AVS3 HDR HFR UHDTV-1 bitstream conformance point defined in clause 5.16.4 in ETSI TS 101 154 [14] (see note 15) |
| | | 0x22 | AVS3 High 10 Profile with resolution up to 7 680 x 4 320 frame rate up to 60 Hz  conformant to the AVS3 HDR UHDTV-2 bitstream conformance point defined in clause 5.16.5 in ETSI TS 101 154 [14] (see note 15) |
| | | 0x23 | AVS3 High 10 Profile with resolution up to 7 680 x 4 320 High Frame Rate of 100 Hz or 120 Hz conformant to the AVS3 HDR HFR UHDTV-2 bitstream conformance point defined in clause 5.16.6 in ETSI TS 101 154 [14] (see note 15) |
| | | 0x24 to 0x2F | reserved for future use for AVS3 |
| | | 0x30 to 0xFF | reserved for future use |
| | 0x1 | 0x00 | AC-4 main audio, mono (see note 12) |
| | | 0x01 | AC-4 main audio, mono, dialogue enhancement enabled (see note 12) |
| | | 0x02 | AC-4 main audio, stereo (see note 12) |
| | | 0x03 | AC-4 main audio, stereo, dialogue enhancement enabled (see note 12) |
| | | 0x04 | AC-4 main audio, multichannel (see note 12) |
| | | 0x05 | AC-4 main audio, multichannel, dialogue enhancement enabled (see note 12) |
| | | 0x06 | AC-4 broadcast-mix audio description, mono, for the visually impaired (see note 12) |
| | | 0x07 | AC-4 broadcast-mix audio description, mono, for the visually impaired, dialogue enhancement enabled (see note 12) |
| | | 0x08 | AC-4 broadcast-mix audio description, stereo, for the visually impaired (see note 12) |
| | | 0x09 | AC-4 broadcast-mix audio description, stereo, for the visually impaired, dialogue enhancement enabled (see note 12) |
| | | 0x0A | AC-4 broadcast-mix audio description, multichannel, for the visually impaired (see note 12) |
| | | 0x0B | AC-4 broadcast-mix audio description, multichannel, for the visually impaired, dialogue enhancement enabled (see note 12) |
| | | 0x0C | AC-4 receiver-mix audio description, mono, for the visually impaired (see note 12) |
| | | 0x0D | AC-4 receiver-mix audio description, stereo, for the visually impaired (see note 12) |
| | | 0x0E | AC-4 Part-2 |
| | | 0x0F | MPEG-H Audio Low Complexity (LC) Profile |

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| | | 0x10 | DTS-UHD main audio, mono (see note 12) |
| | | 0x11 | DTS-UHD main audio, mono, dialogue enhancement enabled (see note 12) |
| | | 0x12 | DTS-UHD main audio, stereo (see note 12) |
| | | 0x13 | DTS-UHD main audio, stereo, dialogue enhancement enabled (see note 12) |
| | | 0x14 | DTS-UHD main audio, multichannel (see note 12) |
| | | 0x15 | DTS-UHD main audio, multichannel, dialogue enhancement enabled (see note 12) |
| | | 0x16 | DTS-UHD broadcast-mix audio description, mono, for the visually impaired (see note 12) |
| | | 0x17 | DTS-UHD broadcast-mix audio description, mono, for the visually impaired, dialogue enhancement enabled (see note 12) |
| | | 0x18 | DTS-UHD broadcast-mix audio description, stereo, for the visually impaired (see note 12) |
| | | 0x19 | DTS-UHD broadcast-mix audio description, stereo, for the visually impaired, dialogue enhancement enabled (see note 12) |
| | | 0x1A | DTS-UHD broadcast-mix audio description, multichannel, for the visually impaired (see note 12) |
| | | 0x1B | DTS-UHD broadcast-mix audio description, multichannel, for the visually impaired, dialogue enhancement enabled (see note 12) |
| | | 0x1C | DTS-UHD receiver-mix audio description, mono, for the visually impaired (see note 12) |
| | | 0x1D | DTS-UHD receiver-mix audio description, stereo, for the visually impaired (see note 12) |
| | | 0x1E | DTS-UHD Next Generation Audio (NGA) Audio |
| | | 0x1F to 0xFF | reserved for future use |
| | 0x2 | 0x00 to 0xFF | Timed Text Markup Language (TTML) subtitles ETSI EN 303 560 [12] |
| | 0x3 to 0xF | 0x00 to 0xFF | reserved for future use |
| 0xA | 0x0 to 0xF | 0x00 to 0xFF | reserved for future use |
| 0xB | 0x0 to 0xD | 0x00 to 0xFF | reserved for future use |
| | 0xE | 0x00 to 0xFF | NGA component type feature flags according to table 27 |
| | 0xF (see note 4) | 0x00 | less than 16:9 aspect ratio |
| | | 0x01 | 16:9 aspect ratio |
| | | 0x02 | greater than 16:9 aspect ratio |
| | | 0x03 | plano-stereoscopic top and bottom (TaB) frame-packing |
| | | 0x04 | HLG10 HDR (see note 4, note 11 and note 13) |
| | | 0x05 | HEVC temporal video subset for a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz (see note 11 and note 13) |
| | | 0x06 | SMPTE ST 2094-10 Dynamic Mapping Information (DMI) format as defined in clause 5.14.4.4.3.4.3 of ETSI TS 101 154 [14] (see note 4 and note 11) |
| | | 0x07 | SL-HDR2 DMI format as defined in clause 5.14.4.4.3.4.4 of ETSI TS 101 154 [14] (see note 4 and note 11) |
| | | 0x08 | SMPTE ST 2094-40 DMI format as defined in clause 5.14.4.4.3.4.5 of ETSI TS 101 154 [14] (see note 4 and note 11) |
| | | 0x09 | PQ10 HDR (see note 4) |
| | | 0x0A to 0xFF | reserved for future use |

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0xC to 0xF | n/a | 0x00 to 0xFF | user defined |
| NOTE 1: | In order to maintain backwards compatibility, the value of the *stream_content_ext* field is not applicable (n/a) for *stream_content* values in the range 0x1 to 0x8, and is set to 0xF. | | |
| NOTE 2: | The terms "standard definition", "high definition", "ultra high definition", "25 Hz", "30 Hz", "50 Hz", and "60 Hz" are used as defined in ETSI TS 101 154 [14], clauses 5.1 to 5.4 for MPEG-2 and clauses 5.5 to 5.7 for H.264/AVC, and clauses 5.14.2 and 5.14.3 for HEVC respectively. The terms "HDR", "HLG10" and "PQ10" are used as defined in clause 5.14.3 of ETSI TS 101 154 [14]. | | |
| NOTE 3: | The specific audio description types indicating the use of broadcast-mix or receiver-mix audio should be preferred over these generic types. For more details see annex J. | | |
| NOTE 4: | These component_descriptor values are intended to be present in combination with one or more component descriptors with the same *component_tag* value. See annex N for example uses. | | |
| NOTE 5: | See ETSI TS 101 547-3 [17] for further information on stereoscopic modes. | | |
| NOTE 6: | 24 Hz video will also use this *component_type*. | | |
| NOTE 7: | Audio streams using AAC audio shall use the corresponding HE AAC values. The AAC profile includes low-complexity AAC. | | |
| NOTE 8: | The specific audio description types indicating the use of broadcast-mix or receiver-mix audio should be preferred over these generic types. For more details see annex J. | | |
| NOTE 9: | For rules on the use of these values, see clause I.2.5 and ETSI TS 101 547-4 [18]. | | |
| NOTE 10: | This value should be used for backward compatible Hybrid Log Gamma with a bit depth of 10 bits (HLG10) HDR services, and/or backward compatible High Frame Rate (HFR) services which are decodable by HEVC_UHDTV_IRD as defined in ETSI TS 101 154 [14]. See also clause I.2.5.2. | | |
| NOTE 11: | For the rules on the use of these values, see clause I.2.6. | | |
| NOTE 12: | These values should be used for elementary streams that convey a single presentation only. | | |
| NOTE 13: | For the rules on the use of these values, see clause I.2.5.2. | | |
| NOTE 14: | For the rules on the use of these values, see clause I.2.8. | | |
| NOTE 15: | For the rules on the use of these values, see clause I.2.9. | | |

Table 27 shows the assignment of *component_type* values in the component_descriptor in the case that the *stream_content* field is set to 0xB and *stream_content_ext* field is set to 0xE. If a bit in the *component_type* is set to 0b1, this indicates that a specific NGA feature (as specified in the description column) is supported. The reserved zero for future use field shall be set to 0b0.

**Table 27: NGA component_type value assignments**

| *component_type* bits | | Description |
|---|---|---|
| b$_7$ (msb) | | reserved_zero_future_use |
| b$_6$ | | content is pre-rendered for consumption with headphones |
| b$_5$ | | content enables interactivity |
| b$_4$ | | content enables dialogue enhancement (see note) |
| b$_3$ | | content contains spoken subtitles |
| b$_2$ | | content contains audio description |
| b$_1$ | b$_0$ | preferred reproduction channel layout |
| 0b0 | 0b0 | no preference |
| 0b0 | 0b1 | stereo |
| 0b1 | 0b0 | two-dimensional |
| 0b1 | 0b1 | three-dimensional |
| NOTE: | Content enabling dialogue enhancement also offers support for clean audio for the hearing impaired. | |

The component_descriptor signalling the available NGA features (*stream_content* 0xB and *stream_content_ext* 0xE) can be used alone without the additional component_descriptor signalling the NGA codec type (*stream_content* 0x9, *stream_content_ext* 0x1, *component_type* 0x0E, 0x0F, or 0x1E), in which case the codec type would be known from *stream_type* and, if present, descriptors within the PMT.

## 6.2.9    Content descriptor

The intention of the content_descriptor (see table 28) is to provide classification information for an event.

**Table 28: Content descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| content_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       content_nibble_level_1 | 4 | uimsbf |
|       content_nibble_level_2 | 4 | uimsbf |
|       user_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the content descriptor:

*content_nibble_level_1*: This 4-bit field represents the first level of a content identifier. This field shall be coded according to table 29.

*content_nibble_level_2*: This 4-bit field represents the second level of a content identifier. This field shall be coded according to table 29.

*user_byte*: This 8-bit field is defined by the broadcaster.

**Table 29: Content nibble level 1 and 2 assignments**

| content_nibble_level_1 | content_nibble_level_2 | Description |
|---|---|---|
| 0x0 | 0x0 to 0xF | undefined content |
| **Movie/Drama:** | | |
| 0x1 | 0x0 | movie/drama (general) |
| 0x1 | 0x1 | detective/thriller |
| 0x1 | 0x2 | adventure/western/war |
| 0x1 | 0x3 | science fiction/fantasy/horror |
| 0x1 | 0x4 | comedy |
| 0x1 | 0x5 | soap/melodrama/folkloric |
| 0x1 | 0x6 | romance |
| 0x1 | 0x7 | serious/classical/religious/historical movie/drama |
| 0x1 | 0x8 | adult movie/drama |
| 0x1 | 0x9 to 0xE | reserved for future use |
| 0x1 | 0xF | user defined |
| **News/Current affairs:** | | |
| 0x2 | 0x0 | news/current affairs (general) |
| 0x2 | 0x1 | news/weather report |
| 0x2 | 0x2 | news magazine |
| 0x2 | 0x3 | documentary |
| 0x2 | 0x4 | discussion/interview/debate |
| 0x2 | 0x5 to 0xE | reserved for future use |
| 0x2 | 0xF | user defined |
| **Show/Game show:** | | |
| 0x3 | 0x0 | show/game show (general) |
| 0x3 | 0x1 | game show/quiz/contest |
| 0x3 | 0x2 | variety show |
| 0x3 | 0x3 | talk show |
| 0x3 | 0x4 to 0xE | reserved for future use |
| 0x3 | 0xF | user defined |
| **Sports:** | | |
| 0x4 | 0x0 | sports (general) |
| 0x4 | 0x1 | special events (Olympic Games, World Cup, etc.) |
| 0x4 | 0x2 | sports magazines |
| 0x4 | 0x3 | football/soccer |
| 0x4 | 0x4 | tennis/squash |
| 0x4 | 0x5 | team sports (excluding football) |
| 0x4 | 0x6 | athletics |
| 0x4 | 0x7 | motor sport |
| 0x4 | 0x8 | water sport |

| content_nibble_level_1 | content_nibble_level_2 | Description |
|---|---|---|
| 0x4 | 0x9 | winter sports |
| 0x4 | 0xA | equestrian |
| 0x4 | 0xB | martial sports |
| 0x4 | 0xC to 0xE | reserved for future use |
| 0x4 | 0xF | user defined |
| **Children's/Youth programmes:** | | |
| 0x5 | 0x0 | children's/youth programmes (general) |
| 0x5 | 0x1 | pre-school children's programmes |
| 0x5 | 0x2 | entertainment programmes for 6 to 14 |
| 0x5 | 0x3 | entertainment programmes for 10 to 16 |
| 0x5 | 0x4 | informational/educational/school programmes |
| 0x5 | 0x5 | cartoons/puppets |
| 0x5 | 0x6 to 0xE | reserved for future use |
| 0x5 | 0xF | user defined |
| **Music/Ballet/Dance:** | | |
| 0x6 | 0x0 | music/ballet/dance (general) |
| 0x6 | 0x1 | rock/pop |
| 0x6 | 0x2 | serious music/classical music |
| 0x6 | 0x3 | folk/traditional music |
| 0x6 | 0x4 | jazz |
| 0x6 | 0x5 | musical/opera |
| 0x6 | 0x6 | ballet |
| 0x6 | 0x7 to 0xE | reserved for future use |
| 0x6 | 0xF | user defined |
| **Arts/Culture (without music):** | | |
| 0x7 | 0x0 | arts/culture (without music, general) |
| 0x7 | 0x1 | performing arts |
| 0x7 | 0x2 | fine arts |
| 0x7 | 0x3 | religion |
| 0x7 | 0x4 | popular culture/traditional arts |
| 0x7 | 0x5 | literature |
| 0x7 | 0x6 | film/cinema |
| 0x7 | 0x7 | experimental film/video |
| 0x7 | 0x8 | broadcasting/press |
| 0x7 | 0x9 | new media |
| 0x7 | 0xA | arts/culture magazines |
| 0x7 | 0xB | fashion |
| 0x7 | 0xC to 0xE | reserved for future use |
| 0x7 | 0xF | user defined |
| **Social/Political issues/Economics:** | | |
| 0x8 | 0x0 | social/political issues/economics (general) |
| 0x8 | 0x1 | magazines/reports/documentary |
| 0x8 | 0x2 | economics/social advisory |
| 0x8 | 0x3 | remarkable people |
| 0x8 | 0x4 to 0xE | reserved for future use |
| 0x8 | 0xF | user defined |
| **Education/Science/Factual topics:** | | |
| 0x9 | 0x0 | education/science/factual topics (general) |
| 0x9 | 0x1 | nature/animals/environment |
| 0x9 | 0x2 | technology/natural sciences |
| 0x9 | 0x3 | medicine/physiology/psychology |
| 0x9 | 0x4 | foreign countries/expeditions |
| 0x9 | 0x5 | social/spiritual sciences |
| 0x9 | 0x6 | further education |
| 0x9 | 0x7 | languages |
| 0x9 | 0x8 to 0xE | reserved for future use |
| 0x9 | 0xF | user defined |
| **Leisure hobbies:** | | |
| 0xA | 0x0 | leisure hobbies (general) |
| 0xA | 0x1 | tourism/travel |
| 0xA | 0x2 | handicraft |
| 0xA | 0x3 | motoring |
| 0xA | 0x4 | fitness and health |

| content_nibble_level_1 | content_nibble_level_2 | Description |
|---|---|---|
| 0xA | 0x5 | cooking |
| 0xA | 0x6 | advertisement/shopping |
| 0xA | 0x7 | gardening |
| 0xA | 0x8 to 0xE | reserved for future use |
| 0xA | 0xF | user defined |
| **Special characteristics:** | | |
| 0xB | 0x0 | original language |
| 0xB | 0x1 | black and white |
| 0xB | 0x2 | unpublished |
| 0xB | 0x3 | live broadcast |
| 0xB | 0x4 | plano-stereoscopic |
| 0xB | 0x5 | local or regional |
| 0xB | 0x6 to 0xE | reserved for future use |
| 0xB | 0xF | user defined |
| **Adult:** | | |
| 0xC | 0x0 | adult (general) |
| 0xC | 0x1 to 0xE | reserved for future use |
| 0xC | 0xF | user defined |
| **Reserved for future use:** | | |
| 0xD to 0xE | 0x0 to 0xF | reserved for future use |
| **User defined:** | | |
| 0xF | 0x0 to 0xF | user defined |

## 6.2.10    Country availability descriptor

In order to identify various combinations of countries efficiently, the descriptor may appear twice for each service, once giving a list of countries and/or groups of countries where the service is intended to be available, and the second giving a list of countries and/or groups where it is not. The latter list overrides the former list. If only one descriptor is used, which lists countries where the service is intended to be available, then it indicates that the service is not intended to be available in any other country. If only one descriptor is used, which lists countries where the service is not intended to be available, then it indicates that the service is intended to be available in every other country. If no descriptor is used, then it is not defined for which countries the service is intended to be available (see table 30).

**Table 30: Country availability descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| country_availability_descriptor() { | | |
|     descriptor_tag | 8 | uimsbf |
|     descriptor_length | 8 | uimsbf |
|     country_availability_flag | 1 | bslbf |
|     reserved_future_use | 7 | bslbf |
|     for (i=0;i<N;i++) { | | |
|         country_code | 24 | bslbf |
|     } | | |
| } | | |

Semantics for the country availability descriptor:

*country_availability_flag*: This 1-bit field indicates whether the following country codes represent the countries in which the reception of the service is intended or not. If *country_availability_flag* is set to 0b1 the following country codes specify the countries in which the reception of the service is intended. If set to 0b0, the following country codes specify the countries in which the reception of the service is not intended.

*country_code*: This 24-bit field identifies a country or group of countries. It shall be coded as three characters, where each character is coded into 8-bits according to ISO/IEC 8859-1 [38] and inserted in order into the field. Individual countries shall be identified by using the alpha-3 code as specified in ISO 3166 [34]. A group of countries shall be identified by the string representation of a decimal number in the range 900 to 999. These numbers shall be coded according to ETSI TS 101 162 [15].

    EXAMPLE:      United Kingdom has 3-character code "GBR", which is coded as:
                    0b0100 0111 0100 0010 0101 0010.

## 6.2.11    Data broadcast descriptor

The `data_broadcast_descriptor` identifies the type of the data component and may be used to provide a text description of the data component (see table 31).

**Table 31: Data broadcast descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| data_broadcast_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    data_broadcast_id | 16 | uimsbf |
|    component_tag | 8 | uimsbf |
|    selector_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       selector_byte | 8 | uimsbf |
|    } | | |
|    ISO_639_language_code | 24 | bslbf |
|    text_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the data broadcast descriptor:

***data_broadcast_id*:** This 16-bit field identifies the data broadcast specification that is used to broadcast the data in the broadcast network. It shall be coded according to ETSI TS 101 162 [15].

***component_tag*:** This 8-bit field has the same value as the *component_tag* field in the `stream_identifier_descriptor` (if present in the PSI `program_map_section`) for the component stream. If this field is not used it shall be set to the value 0x00.

***selector_length*:** This 8-bit field specifies the length in bytes of the following selector field.

***selector_byte*:** This is an 8-bit field. The sequence of *selector_byte* fields specifies the selector field. The syntax and semantics of the selector field shall be defined by the data broadcast specification that is identified in the *data_broadcast_id* field. The selector field may contain service specific information that is necessary to identify an entry-point of the broadcast data.

***ISO_639_language_code*:** This 24-bit field identifies the language of the following text field. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

    EXAMPLE:    French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101

***text_length*:** This 8-bit field specifies the length in bytes of the following text describing the data component.

***char*:** This is an 8-bit field, a sequence of which conveys a text description of the data component. Text information is coded using the character sets and methods described in annex A.

## 6.2.12    Data broadcast id descriptor

The `data_broadcast_id_descriptor` identifies the type of the data component (see table 32). It is a short form of the `data_broadcast_descriptor` and it may be placed in the component loop of the PMT table.

**Table 32: Data broadcast id descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| data_broadcast_id_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    data_broadcast_id | 16 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       selector_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the data broadcast id descriptor:

*data_broadcast_id***:** This 16-bit field identifies the data broadcast specification that is used to broadcast the data in the broadcast network. It shall be coded according to ETSI TS 101 162 [15].

*selector_byte***:** This is an 8-bit field. The sequence of *selector_byte* fields specifies the selector field. For the purpose of application selection the selector field might be used. The definition of the selector field of the data_broadcast_id_descriptor will depend on the *data_broadcast_id*.

> NOTE: The contents of the selector field in a data_broadcast_id_descriptor may differ from the contents of the selector field of a corresponding data_broadcast_descriptor with the same *data_broadcast_id*.

## 6.2.13 Delivery system descriptors

### 6.2.13.1 Cable delivery system descriptor

See table 33.

**Table 33: Cable delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| cable_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    frequency | 32 | bslbf |
|    reserved_future_use | 12 | bslbf |
|    FEC_outer | 4 | bslbf |
|    modulation | 8 | bslbf |
|    symbol_rate | 28 | bslbf |
|    FEC_inner | 4 | bslbf |
| } | | |

Semantics for the cable delivery system descriptor:

*frequency***:** The frequency is a 32-bit field giving the 4-bit BCD values specifying 8 characters of the frequency value. For the cable_delivery_system_descriptor, the frequency is coded in MHz, where the decimal occurs after the fourth character.

> EXAMPLE 1: A *frequency* value of 0x03120000 indicates a frequency of 312,0000 MHz.

*FEC_outer***:** This is a 4-bit field specifying the outer Forward Error Correction (FEC) scheme used according to table 34.

**Table 34: Outer FEC scheme**

| *FEC_outer* | Description |
|---|---|
| 0b0000 | not defined |
| 0b0001 | no outer FEC coding |
| 0b0010 | (204,188) Reed-Solomon code (RS) |
| 0b0011 to 0b1111 | reserved for future use |

*modulation*: This is an 8-bit field. It specifies the modulation scheme used on a cable delivery system according to table 35.

**Table 35: Modulation scheme for cable**

| modulation | Description |
|---|---|
| 0x00 | not defined |
| 0x01 | 16-ary Quadrature Amplitude Modulation (16QAM) |
| 0x02 | 32-ary Quadrature Amplitude Modulation (32QAM) |
| 0x03 | 64-ary Quadrature Amplitude Modulation (64QAM) |
| 0x04 | 128-ary Quadrature Amplitude Modulation (128QAM) |
| 0x05 | 256-ary Quadrature Amplitude Modulation (256QAM) |
| 0x06 to 0xFF | reserved for future use |

*symbol_rate*: This is a 28-bit field giving the 4-bit BCD values specifying 7 characters of the symbol rate in Msymbol/s where the decimal point occurs after the third character.

   EXAMPLE 2:    A *symbol_rate* value of 0x0274500 indicates a symbol rate of 27,4500 Msymbol/s.

*FEC_inner*: This is a 4-bit field specifying the inner FEC scheme used according to table 36.

**Table 36: Inner FEC scheme**

| FEC_inner (see note) | Description |
|---|---|
| 0b0000 | not defined |
| 0b0001 | 1/2 convolutional code rate |
| 0b0010 | 2/3 convolutional code rate |
| 0b0011 | 3/4 convolutional code rate |
| 0b0100 | 5/6 convolutional code rate |
| 0b0101 | 7/8 convolutional code rate |
| 0b0110 | 8/9 convolutional code rate |
| 0b0111 | 3/5 convolutional code rate |
| 0b1000 | 4/5 convolutional code rate |
| 0b1001 | 9/10 convolutional code rate |
| 0b1010 to 0b1110 | reserved for future use |
| 0b1111 | no convolutional coding |
| NOTE:      Not all convolutional code rates apply for all modulation schemes. ||

## 6.2.13.2      Satellite delivery system descriptor

See table 37.

**Table 37: Satellite delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| satellite_delivery_system_descriptor() { | | |
|     descriptor_tag | 8 | uimsbf |
|     descriptor_length | 8 | uimsbf |
|     frequency | 32 | bslbf |
|     orbital_position | 16 | bslbf |
|     west_east_flag | 1 | bslbf |
|     polarization | 2 | bslbf |
|     if (modulation_system == 0b1) { | | |
|         roll_off | 2 | bslbf |
|     } else { | | |
|         reserved_zero_future_use | 2 | bslbf |
|     } | | |
|     modulation_system | 1 | bslbf |
|     modulation_type | 2 | bslbf |
|     symbol_rate | 28 | bslbf |
|     FEC_inner | 4 | bslbf |
| } | | |

Semantics for the satellite delivery system descriptor:

*frequency***:** The frequency is a 32-bit field giving the 4-bit BCD values specifying 8 characters of the frequency value. For the `satellite_delivery_system_descriptor`, the frequency is coded in GHz, where the decimal point occurs after the third character.

    EXAMPLE 1:    A *frequency* value of 0x01175725 indicates a frequency of 11,75725 GHz.

*orbital_position***:** The *orbital_position* is a 16-bit field giving the 4-bit BCD values specifying 4 characters of the orbital position in degrees where the decimal point occurs after the third character (e.g. 019,2°).

*west_east_flag***:** The *west_east_flag* is a 1-bit field indicating if the satellite position is in the western or eastern part of the orbit. A value 0b0 indicates the western position and a value 0b1 indicates the eastern position.

*polarization***:** The *polarization* is a 2-bit field specifying the polarization of the transmitted signal. The first bit defines whether the polarization is linear or circular (see table 38).

**Table 38: Polarization coding**

| *polarization* | Description |
|---|---|
| 0b00 | linear - horizontal |
| 0b01 | linear - vertical |
| 0b10 | circular - left |
| 0b11 | circular - right |

*roll_off***:** This 2-bit field specifies the roll-off factor used in DVB-S2. It shall be coded according to table 39.

**Table 39: Roll-off factor**

| *roll_off* | Description |
|---|---|
| 0b00 | $\alpha = 0{,}35$ |
| 0b01 | $\alpha = 0{,}25$ |
| 0b10 | $\alpha = 0{,}20$ |
| 0b11 | reserved for future use |

*modulation_system***:** This is a 1-bit field. It specifies the broadcast scheme used on a satellite delivery system according to table 40.

If DVB Satellite Framing and Modulation, Second Generation (DVB-S2) is used with scrambling sequences other than the default in physical layer scrambling, or with multiple input streams, or with time slicing, the `S2_satellite_delivery_system_descriptor` shall be used additionally (see clause 6.2.13.3).

**Table 40: Modulation system for satellite**

| *modulation_system* | Description |
|---|---|
| 0b0 | DVB-S |
| 0b1 | DVB-S2 |

*modulation_type***:** This is a 2-bit field. It specifies the modulation scheme used on a satellite delivery system according to table 41.

**Table 41: Modulation type for satellite**

| *modulation_type* | Description |
|---|---|
| 0b00 | auto |
| 0b01 | Quaternary Phase Shift Keying (QPSK) |
| 0b10 | 8-ary Phase Shift Keying (8PSK) |
| 0b11 | 16QAM (n/a for DVB-S2) |

*symbol_rate***:** This is a 28-bit field giving the 4-bit BCD values specifying 7 characters of the symbol rate in Msymbol/s where the decimal point occurs after the third character.

EXAMPLE 2: A *symbol_rate* value of 0x0274500 indicates a symbol rate of 27,4500 Msymbol/s.

*FEC_inner*: This is a 4-bit field specifying the inner FEC scheme used according to table 36.

## 6.2.13.3    S2 satellite delivery system descriptor

When DVB-S2 is used, this descriptor (see table 42) may be used in addition to the
satellite_delivery_system_descriptor (see clause 6.2.13.2) to indicate the scrambling sequence used in physical
layer scrambling, or the use of multiple input streams, or the use of time slicing.

**Table 42: S2 satellite delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| S2_satellite_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    scrambling_sequence_selector | 1 | bslbf |
|    multiple_input_stream_flag | 1 | bslbf |
|    reserved_zero_future_use | 1 | bslbf |
|    not_timeslice_flag | 1 | bslbf |
|    reserved_future_use | 2 | bslbf |
|    TS_GS_mode | 2 | uimsbf |
|    if (scrambling_sequence_selector == 0b1) { | | |
|       reserved_future_use | 6 | bslbf |
|       scrambling_sequence_index | 18 | uimsbf |
|    } | | |
|    if (multiple_input_stream_flag == 0b1) { | | |
|       input_stream_identifier | 8 | uimsbf |
|    } | | |
|    if (not_timeslice_flag == 0b0) { | | |
|       timeslice_number | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the S2 satellite delivery system descriptor:

*scrambling_sequence_selector*: A value 0b0 specifies that the default DVB-S2 physical layer scrambling sequence of
index n=0 is used. A value 0b1 means that the default scrambling sequence is not used and that the sequence index n is
conveyed by the *scrambling_sequence_index* field.

*multiple_input_stream_flag*: The *multiple_input_stream_flag* is a 1-bit field indicating whether a single or multiple
DVB transport streams are conveyed. A value 0b0 indicates that a single DVB transport stream is carried. A value 0b1
indicates that multiple DVB transport streams are conveyed and that the *input_stream_identifier* field is present.

*TS_GS_mode*: This 2-bit field indicates which mode the DVB-S2 Input Stream is operating in. It shall be coded
according to table 43.

**Table 43: Coding of the TS GS mode**

| *TS_GS_mode* (see note) | Description |
|---|---|
| 0 | Generic Packetized |
| 1 | Generic Stream Encapsulation (GSE) |
| 2 | DVB transport stream |
| 3 | reserved for future use |
| NOTE: | These values are different from similar assignments in table 3 in ETSI EN 302 307-1 [7]. |

*scrambling_sequence_index*: This 18-bit field, when present, carries the index of the DVB-S2 physical layer
scrambling sequence as defined in clause 5.5.4 of ETSI EN 302 307-1 [7].

*not_timeslice_flag*: This 1-bit field indicates the use of time slicing as defined in annex M of ETSI EN 302 307-1 [7].
When set to 0b1, it indicates that time slicing is not used. When set to 0b0, it indicates that time slicing is used.

*input_stream_identifier***:** This 8-bit field carries the DVB-S2 Input Stream Identifier (ISI) as defined in clause 5.1.6 of ETSI EN 302 307-1 [7].

*timeslice_number***:** When present, this 8-bit field indicates which time slice carries all data for the stream being described.

### 6.2.13.4      Terrestrial delivery system descriptor

See table 44.

> NOTE:      The delivery system descriptor for DVB Terrestrial Framing and Modulation, Second Generation
> (DVB-T2) transmissions is specified in clause 6.4.6.3.

**Table 44: Terrestrial delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| terrestrial_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    centre_frequency | 32 | uimsbf |
|    bandwidth | 3 | bslbf |
|    priority | 1 | bslbf |
|    time_slicing_indicator | 1 | bslbf |
|    MPE-FEC_indicator | 1 | bslbf |
|    reserved_future_use | 2 | bslbf |
|    constellation | 2 | bslbf |
|    hierarchy_information | 3 | bslbf |
|    code_rate_HP_stream | 3 | bslbf |
|    code_rate_LP_stream | 3 | bslbf |
|    guard_interval | 2 | bslbf |
|    transmission_mode | 2 | bslbf |
|    other_frequency_flag | 1 | bslbf |
|    reserved_future_use | 32 | bslbf |
| } | | |

Semantics for the terrestrial delivery system descriptor:

*centre_frequency***:** This 32-bit field indicates the centre frequency value in multiples of 10 Hz. The coding range is from minimum 10 Hz (0x0000 0001) up to a maximum of 42 949 672 950 Hz (0xFFFF FFFF).

*bandwidth***:** This is a 3-bit field specifying the bandwidth in use. It shall be coded according to table 45.

**Table 45: Bandwidth coding**

| *bandwidth* | Description |
|---|---|
| 0b000 | 8 MHz |
| 0b001 | 7 MHz |
| 0b010 | 6 MHz |
| 0b011 | 5 MHz |
| 0b100 to 0b111 | reserved for future use |

*priority***:** This 1-bit flag indicates the stream's hierarchical priority. It shall be coded according to table 46. In case the *hierarchy_information* field is not equal to 0b000, if *priority* is set to 0b1, it indicates that the associated DVB transport stream is a High Priority (HP) stream, and if *priority* is set to 0b0, the associated DVB transport stream is a Low Priority (LP) stream. In case the *hierarchy_information* field has the value 0b000, the priority flag shall be set to 0b1.

**Table 46: Priority coding**

| *priority* | Description |
|---|---|
| 0b0 | HP |
| 0b1 | LP |

*time_slicing_indicator*: This 1-bit field indicates the use of the time slicing on the associated DVB transport stream. If the *time_slicing_indicator* is set (0b1), time slicing is not used. If the *time_slicing_indicator* is cleared (set to 0b0), at least one elementary stream uses time slicing.

Time slicing is defined in ETSI EN 301 192 [3].

*MPE-FEC_indicator*: This 1-bit field indicates the use of the Multi-Protocol Encapsulation (MPE)-FEC on the associated DVB transport stream. If the *MPE-FEC_indicator* is set (0b1), MPE-FEC is not used. If the *MPE-FEC_indicator* is cleared (0b0), at least one elementary stream uses MPE-FEC.

MPE-FEC is defined in ETSI EN 301 192 [3].

*constellation*: This is a 2-bit field. It specifies the constellation pattern used on a terrestrial delivery system according to table 47.

**Table 47: Constellation coding**

| constellation | Description |
|---|---|
| 0b00 | QPSK |
| 0b01 | 16QAM |
| 0b10 | 64QAM |
| 0b11 | reserved for future use |

*hierarchy_information*: The *hierarchy_information* specifies whether the transmission is hierarchical and, if so, what the a value is (see table 48). Also, the use of in-depth interleaver is indicated. When the *transmission_mode* indicates the use of 8k mode, only the native interleaver shall be signalled.

**Table 48: Hierarchy information coding**

| hierarchy_information | Description |
|---|---|
| 0b000 | non-hierarchical, native interleaver |
| 0b001 | $\alpha = 1$, native interleaver |
| 0b010 | $\alpha = 2$, native interleaver |
| 0b011 | $\alpha = 4$, native interleaver |
| 0b100 | non-hierarchical, in-depth interleaver |
| 0b101 | $\alpha = 1$, in-depth interleaver |
| 0b110 | $\alpha = 2$, in-depth interleaver |
| 0b111 | $\alpha = 4$, in-depth interleaver |

*code_rate_HP_stream* and *code_rate_LP_stream*: These 3-bit fields specify the inner FEC scheme according to table 49, which is used by the HP and LP stream, respectively. Non-hierarchical channel coding and modulation requires signalling of one code rate. In this case, 3 bits specifying the code rate according to table 49 (*code_rate_HP_stream*) are followed by another 3 bits of value 0b000 (*code_rate_LP_stream*). Two different code rates may be applied to two different levels of modulation with the aim of achieving hierarchy. Transmission then starts with the code rate for the HP level of the modulation and ends with the one for the LP level.

**Table 49: HP and LP stream code rate coding**

| code_rate_HP_stream and code_rate_LP_stream | Description |
|---|---|
| 0b000 | 1/2 |
| 0b001 | 2/3 |
| 0b010 | 3/4 |
| 0b011 | 5/6 |
| 0b100 | 7/8 |
| 0b101 to 0b111 | reserved for future use |

*guard_interval*: The *guard_interval* is a 2-bit field specifying the guard interval according to table 50.

**Table 50: Guard interval coding**

| guard_interval | Description |
|:---:|:---:|
| 0b00 | 1/32 |
| 0b01 | 1/16 |
| 0b10 | 1/8 |
| 0b11 | 1/4 |

*transmission_mode***:** This 2-bit field indicates the number of carriers in an Orthogonal Frequency Division Multiplex (OFDM) frame according to table 51.

**Table 51: Transmission mode coding**

| transmission_mode | Description |
|:---:|:---|
| 0b00 | 2k mode |
| 0b01 | 8k mode |
| 0b10 | 4k mode |
| 0b11 | reserved for future use |

*other_frequency_flag***:** This 1-bit flag indicates whether other frequencies are in use. The value 0b0 indicates that no other frequency is in use, 0b1 indicates that one or more other frequencies are in use.

## 6.2.14   DSNG descriptor

In Digital Satellite News Gathering (DSNG) transmissions the TSDT shall be present in the bitstream and the TSDT descriptor loop shall contain the DSNG_descriptor with the American Standard Code for Information Interchange (ASCII) codes for "CONA" in the text field (see ETSI EN 301 210 [4]).

For DSNG applications at least one DSNG_descriptor (see table 52) shall be present in the TSDT.

**Table 52: DSNG descriptor**

| Syntax | Number of bits | Identifier |
|:---|:---:|:---:|
| DSNG_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the DSNG descriptor:

*byte***:** The sequence of bytes in the DSNG descriptor is defined in ETSI EN 301 210 [4].

## 6.2.15   Extended event descriptor

The extended_event_descriptor (see table 53) provides a detailed text description of an event, which may be used in addition to the short_event_descriptor. More than one extended_event_descriptor can be associated to allow information about one event greater in length than 256 bytes to be conveyed. Text information can be structured into two columns, one giving an item description field and the other the item text. A typical application for this structure is to give a cast list, where for example the item description field might be "Producer" and the item field would give the name of the producer.

**Table 53: Extended event descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| extended_event_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| descriptor_number | 4 | uimsbf |
| last_descriptor_number | 4 | uimsbf |
| ISO_639_language_code | 24 | bslbf |
| length_of_items | 8 | uimsbf |
| for (i=0;i<N;i++) { | | |
| item_description_length | 8 | uimsbf |
| for (j=0;j<N;j++) { | | |
| item_description_char | 8 | uimsbf |
| } | | |
| item_length | 8 | uimsbf |
| for (j=0;j<N;j++) { | | |
| item_char | 8 | uimsbf |
| } | | |
| } | | |
| text_length | 8 | uimsbf |
| for (i=0;i<N;i++) { | | |
| text_char | 8 | uimsbf |
| } | | |
| } | | |

Semantics for the extended event descriptor:

*descriptor_number*: This 4-bit field gives the number of the descriptor. It is used to associate information which cannot be fitted into a single descriptor. The *descriptor_number* of the first of an associated set of descriptors shall be 0x00. The *descriptor_number* shall be incremented by 1 with each additional extended_event_descriptor in this section.

*last_descriptor_number*: This 4-bit field specifies the number of the last extended_event_descriptor (that is, the descriptor with the highest value of *descriptor_number*) of the associated set of descriptors of which this descriptor is part.

*ISO_639_language_code*: This 24-bit field identifies the language of the following text fields. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

    EXAMPLE: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*length_of_items*: This is an 8-bit field specifying the length in bytes of the following items.

*item_description_length*: This 8-bit field specifies the length in bytes of the item description.

*item_description_char*: This is an 8-bit field, a sequence of which conveys the item description. Text information is coded using the character sets and methods described in annex A.

*item_length*: This 8-bit field specifies the length in bytes of the item text.

*item_char*: This is an 8-bit field, a sequence of which conveys the item text. Text information is coded using the character sets and methods described in annex A.

*text_length*: This 8-bit field specifies the length in bytes of the non-itemized extended text.

*text_char*: This is an 8-bit field, a sequence of which conveys the non-itemized extended text. Text information is coded using the character sets and methods described in annex A.

## 6.2.16 Extension descriptor

The extension_descriptor (see table 54) is used to extend the 8-bit namespace of the *descriptor_tag* field.

**Table 54: Extension descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| extension_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| descriptor_tag_extension | 8 | uimsbf |
| for (i=0;i<N;i++) { | | |
| selector_byte | 8 | bslbf |
| } | | |
| } | | |

Semantics for the extension descriptor:

*descriptor_tag_extension***:** The *descriptor_tag_extension* is an 8-bit field which identifies each extended descriptor. The values of *descriptor_tag_extension* are defined in clause 6.3.

*selector_byte***:** This is an 8-bit field. The sequence of *selector_byte* fields specifies the selector field. The syntax and semantics of the selector field are defined in clause 6.4.

## 6.2.17   Frequency list descriptor

The frequency_list_descriptor (see table 55) may be used in the NIT. It gives the complete list of additional frequencies for a certain multiplex which is transmitted on multiple frequencies.

**Table 55: Frequency list descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| frequency_list_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| reserved_future_use | 6 | bslbf |
| coding_type | 2 | bslbf |
| for (i=0;i<N;i++) { | | |
| centre_frequency | 32 | uimsbf |
| } | | |
| } | | |

Semantics for the frequency list descriptor:

*coding_type***:** This is a 2-bit field that indicates how the frequency is coded and relates to the delivery system used. It has a value indicated in table 56.

**Table 56: Coding type coding**

| coding_type | Description |
|---|---|
| 0b00 | not defined |
| 0b01 | satellite |
| 0b10 | cable |
| 0b11 | terrestrial |

*centre_frequency***:** This is as defined in the delivery system descriptor for the delivery system given by the *coding_type*.

## 6.2.18   FTA content management descriptor

### 6.2.18.1    Semantics and syntax of the FTA content management descriptor

The FTA content management descriptor (see table 57) provides a means of defining the content management policy for an item of content delivered as part of a Free-To-Air (FTA) DVB service.

The signalling conveyed by this descriptor has been designed for use in the context of the DVB's CPCM solution. However, the descriptor may also be used in a non-CPCM context.

The FTA flags are intended to signal to a receiving device (not limited to a DVB CPCM implementation) the configuration of internal states that affect the means by which content might be redistributed. It reflects the broadcaster's intention for content usage and shall result in a similar user experience across implementations.

The FTA content usage restrictions defined in the present clause apply to "remote access over the Internet" and "content scrambling". They can be summarized as follows:

1) When remote access is fully enabled (*control_remote_access_over_internet* = 0b00), content can be viewed, copied, moved and remotely accessed.

2) When remote access is enabled within a managed domain (*control_remote_access_over_internet* = 0b01) content can be viewed, copied and moved locally only; and accessed remotely within a managed domain.

3) If remote access is enabled within a managed domain (*control_remote_access_over_internet* = 0b10) only after 24 hours of the original broadcast to protect e.g. live transmissions, content can be viewed, copied and moved locally from the time of reception until 24 hours after reception; and then after the 24 hour delay time has expired may additionally be accessed remotely within a managed domain.

4) When remote access is not permitted (*control_remote_access_over_internet* = 0b11) content can be viewed, copied and moved locally only. Some content protection systems may be permitted to expire this prohibition of remote access within a managed domain after a very long time defined by their compliance regimes.

NOTE 1: In revisions of the present document prior to V1.17.1, this ordered list used letters ("a)", "b)", and so on) to label list items. For technical reasons, starting with revision 1.17.1 list numbering now uses numbers as item labels ("1)", "2)", and so on). As the order of list items was not altered when introducing this change, references to letter-labelled list items can directly and unambiguously be converted to number-labelled list items by mapping the ordinal position of the letter-labelled item in the lexicographical ordering, to the same ordinal position in the numerical ordering of the number-labelled items (i.e. "a)" corresponds to "1)", "b)" corresponds to "2)" and so on).

For cases 2), 3) and 4), a managed domain is a distinguishable set of devices belonging to the same household, within which content usage can be controlled. This definition is subject to further refinement by compliance regimes. Likewise, "local" means within the same immediate vicinity, approximating to the physical extent of a domicile or vehicle. This implies the need for devices to determine whether a device is local to another device prior to applying the related usage rules.

For cases 2), 3) and 4), the transfer of content from/to removable storage media is by default authorized locally. This removable storage media can be moved to another location and played, even if the playback device belongs to a different managed domain than the one from which the content was transferred to this removable storage media. If content on the removable media storage is bound to the managed domain, content can be further moved and copied within the managed domain. Otherwise, the copy of content shall be bound to the removable storage media and marked "copy no more" once on the removable storage media.

By default, the signalling does not call for downscaling of content output over analogue interfaces.

When the scrambling after acquisition is enabled (*do_not_scramble* = 0b0), digital links shall be protected (e.g. Digital Transmission Content Protection (DTCP), High-bandwidth Digital Content Protection (HDCP)).

The FTA signalling information should remain with, and associated to, the content during its entire lifetime as a reminder of the broadcaster's intention.

Implementers are advised to consult possible additional usage and implementation restrictions (e.g. market/regulatory specific), which are not defined in the present document.

**Table 57: FTA content management descriptor**

| Syntax | Number of bits | Identifier |
|--------|:--------------:|:----------:|
| FTA_content_management_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    user_defined | 1 | bslbf |
|    reserved_future_use | 3 | bslbf |
|    do_not_scramble | 1 | uimsbf |
|    control_remote_access_over_internet | 2 | uimsbf |
|    do_not_apply_revocation | 1 | uimsbf |
| } | | |

Semantics for the FTA content management descriptor:

*user_defined*: The semantics of this 1-bit field may be defined in specifications outside the scope of the present document. If no semantics are defined, it shall be set to zero.

> NOTE 2: Since this flag appears inside a descriptor dealing with restrictions applicable to the usage of content, misinterpretation by the receiver should be avoided. Broadcasters intending to make use of this bit are thus encouraged to coordinate its use with other broadcasters and manufacturers. For Europe, use of this bit is coordinated by DVB.

*do_not_scramble*: This is a 1-bit field that indicates whether or not to apply scrambling of the content item for the purposes of content protection.

If *do_not_scramble* is set to 0b1 then scrambling shall not be applied for the purposes of content protection. If *do_not_scramble* is set to 0b0 then scrambling shall be applied where applicable for content protection.

The specification of "where applicable" is not defined by the present document.

*control_remote_access_over_internet*: This is a 2-bit field. It indicates the policy regarding redistribution of the content item over the internet according to table 58.

**Table 58: Coding of control_remote_access_over_internet**

| *control_remote_access_over_internet* | Description |
|:---:|---|
| 0b00 | Redistribution over the Internet is enabled. |
| 0b01 | Redistribution over the Internet is enabled but only within a managed domain. |
| 0b10 | Redistribution over the Internet is enabled but only within a managed domain and after a certain short period of time (e.g. 24 hours). |
| 0b11 | Redistribution over the Internet is not allowed with the following exception: Redistribution over the Internet within a managed domain is enabled after a specified long (possibly indefinite) period of time. |

In the absence of a managed domain, the values 0b01 and 0b10 shall be interpreted as 0b11. Hence, since there is no managed domain, this means redistribution over the internet is not allowed.

The specification of "redistribution over the Internet", "managed domain", "certain short period of time" and "specified long (possibly indefinite) period of time" are not defined by the present document.

*do_not_apply_revocation*: This 1-bit field indicates whether or not to apply the content revocation process to the content item.

If *do_not_apply_revocation* is set to 0b1 then the content revocation process shall not be applied. If *do_not_apply_revocation* is set to 0b0 then the content revocation process shall be applied.

The specification of "content revocation process" is not defined by the present document.

## 6.2.18.2          Scope of the FTA content management descriptor

The location of a particular instance of the descriptor defines the scope over which the defined content management policy shall apply. This policy shall apply to all items of content within this scope unless overridden at a scope of greater precedence.

The content management policy for an item of content that falls outside of the scope of all explicit signalling is not defined by the present document.

The following scoping rules are listed in order of increasing precedence, each overriding the scope of the previous one:

- To define a content management policy with the scope of all DVB services within a network, a single instance may be placed in the corresponding first descriptor loop of the NIT sub_table. If a NIT sub_table contains multiple sections then the descriptor shall only appear in the first descriptor loop of one section.

- To define a content management policy with the scope of all DVB services within a bouquet, a single instance may be placed in the corresponding first descriptor loop of the BAT sub_table. If a BAT sub_table contains multiple sections then the descriptor shall only appear in the first descriptor loop of one section.

- To define a content management policy with the scope of all DVB services within a DVB transport stream of a particular network, a single instance may be placed in the corresponding transport stream descriptor loop of the NIT. The content management policy for a DVB transport stream overrides the content management policy of the network (if defined).

- To define a content management policy with the scope of all DVB services within a DVB transport stream of a particular bouquet, a single instance may be placed in the corresponding transport stream descriptor loop of the BAT. The content management policy for a DVB transport stream overrides the content management policy of the bouquet (if defined).

- To define a content management policy for a single DVB service a single instance may be placed in the corresponding descriptor loop of the SDT. The content management policy for a service overrides the content management policy of the DVB transport stream, bouquet, or network (if defined).

- A content management policy for a single event within a DVB service can only be defined for the present event. A single instance may be placed in the descriptor loop corresponding to the present event in EIT present/following. The content management policy for an event overrides the content management policy of the service, DVB transport stream, bouquet or network (if defined).

NOTE:          A single instance may also be placed in each of the descriptor loops of an EIT sub_table but this will only be considered as informative for anything other than the present event of EIT present/following.

The effect of defining a content management policy in a BAT that conflicts with a definition of equivalent scope in a NIT is not defined by the present document.

## 6.2.19          Linkage descriptor

### 6.2.19.1          Semantics and syntax of the linkage descriptor

The `linkage_descriptor` (see table 59) identifies a service that can be presented if the consumer requests for additional information related to a specific entity described by the SI system. The location of the `linkage_descriptor` in the syntax indicates the entity for which additional information is available. For example a `linkage_descriptor` located within the NIT shall point to a service providing additional information on the network, a `linkage_descriptor` in the BAT shall provide a link to a service informing about the bouquet, etc.

A CA replacement service can be identified using the `linkage_descriptor`. This service may be selected automatically by the IRD if the CA denies access to the specific entity described by the SI system.

A service replacement service can also be identified using the `linkage_descriptor`. This replacement service may be selected automatically by the IRD when the running status of the current service is set to "not running".

**Table 59: Linkage descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| linkage_descriptor() {<br>   descriptor_tag<br>   descriptor_length<br>   transport_stream_id<br>   original_network_id<br>   service_id<br>   linkage_type<br>   if (linkage_type == 0x08) {<br>      mobile_hand-over_info()<br>   } else if (linkage_type == 0x0D) {<br>      event_linkage_info()<br>   } else if (linkage_type >= 0x0E && linkage_type <= 0x1F) {<br>      extended_event_linkage_info()<br>   }<br>   for (i=0;i<N;i++) {<br>      private_data_byte<br>   }<br>} | <br>8<br>8<br>16<br>16<br>16<br>8<br><br><br><br><br><br><br><br><br><br><br><br>8 | <br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br><br><br><br><br><br><br><br><br><br><br><br>bslbf |

Semantics for the linkage descriptor:

*transport_stream_id***:** This is a 16-bit field which identifies the TS containing the information service indicated.

*original_network_id***:** This 16-bit field gives the label identifying the *network_id* of the originating delivery system of the information service indicated.

*service_id***:** This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding program_map_section, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (NVOD reference services) the *service_id* does not have a corresponding *program_number*.

If the *linkage_type* field has the value 0x04, then the *service_id* field is not relevant, and shall be set to 0x0000.

*linkage_type***:** This is an 8-bit field specifying the type of linkage e.g. to information (see table 60).

**Table 60: Linkage type coding**

| *linkage_type* | Description |
|---|---|
| 0x00 | reserved for future use |
| 0x01 | information service |
| 0x02 | EPG service |
| 0x03 | CA replacement service |
| 0x04 | TS containing complete network/bouquet SI |
| 0x05 | service replacement service |
| 0x06 | data broadcast service |
| 0x07 | Return Channel Satellite (RCS) map |
| 0x08 | mobile hand-over |
| 0x09 | System Software Update (SSU) service (ETSI TS 102 006 [20]) |
| 0x0A | TS containing SSU BAT or NIT (ETSI TS 102 006 [20]) |
| 0x0B | Internet Protocol/Medium Access Control (IP/MAC) notification service (ETSI EN 301 192 [3]) |
| 0x0C | TS containing INT BAT or NIT (ETSI EN 301 192 [3]) |
| 0x0D | event linkage (see note) |
| 0x0E to 0x1F | extended event linkage (see note) |
| 0x20 | downloadable font info linkage (ETSI EN 303 560 [12]) |
| 0x21 | Native IP bootstrap MPE stream (DVB BlueBook A180 [57]) |
| 0x22 to 0x7F | reserved for future use |
| 0x80 to 0xFE | user defined |
| 0xFF | reserved for future use |
| NOTE: A *linkage_type* with a value in the range 0x0D to 0x1F is only valid when the descriptor is carried in the EIT. | |

`mobile_hand-over_info():` This field shall be coded according to clause 6.2.19.2.

`event_linkage_info():` This field shall be coded according to clause 6.2.19.3.

`extended_event_linkage_info():` This field shall be coded according to clause 6.2.19.4.

*private_data_byte*: This is an 8-bit field, the value of which is privately defined.

## 6.2.19.2    Mobile hand-over linkage

A service to which a mobile receiver might hand-over to can also be identified using the `mobile_hand-over_info` structure in the `linkage_descriptor`. This service may be selected automatically by the IRD when the actual service is no longer receivable under its *service_id*. The *hand-over_type* identifies whether the `linkage_descriptor` links to the same service in a different country, to a local variation of the service, or an associated service.

The `mobile_hand-over_info` shall be coded according to table 61.

**Table 61: Mobile hand-over info**

| Syntax | Number of bits | Identifier |
|---|---|---|
| mobile_hand-over_info() { | | |
|    hand-over_type | 4 | uimsbf |
|    reserved_future_use | 3 | bslbf |
|    origin_type | 1 | bslbf |
|    if (hand-over_type == 0x1 | | |
|      \|\| hand-over_type == 0x2 | | |
|      \|\| hand-over_type == 0x3) { | | |
|      network_id | 16 | uimsbf |
|    } | | |
|    if (origin_type == 0b0) { | | |
|      initial_service_id | 16 | uimsbf |
|    } | | |
| } | | |

Semantics for the mobile hand-over info:

*hand-over_type*: This is a 4-bit field specifying the type of hand-over (see table 62).

**Table 62: Hand-over type coding**

| *hand-over_type* | Description |
|---|---|
| 0x0 | reserved for future use |
| 0x1 | DVB hand-over to an identical service in a neighbouring country |
| 0x2 | DVB hand-over to a local variation of the same service |
| 0x3 | DVB hand-over to an associated service |
| 0x4 to 0xF | reserved for future use |

*origin_type*: This is a flag specifying in which table the link is originated (see table 63).

**Table 63: Origin type coding**

| *origin_type* | Description |
|---|---|
| 0b0 | NIT |
| 0b1 | SDT |

*network_id*: This is a 16-bit field which identifies the terrestrial network that supports the service indicated.

*initial_service_id*: This is a 16-bit field which identifies the service for which the hand-over linkage is valid.

### 6.2.19.3    Event linkage

Two events can be signalled as equivalent using the using the `event_linkage_info` structure in the `linkage_descriptor`. The event being linked to may be a simulcast or may be time offset. The *event_simulcast* flag shall only be set if the target event is higher quality.

The `event_linkage_info` shall be coded according to table 64.

**Table 64: Event linkage info**

| Syntax | Number of bits | Identifier |
|---|---|---|
| event_linkage_info() { | | |
|     target_event_id | 16 | uimsbf |
|     target_listed | 1 | bslbf |
|     event_simulcast | 1 | bslbf |
|     reserved_future_use | 6 | bslbf |
| } | | |

Semantics for the event linkage info:

**target_event_id:** This 16-bit field identifies the *event_id* of the event (the target event), carried on the service defined by the *original_network_id*, *transport_stream_id*, and *service_id*, which is equivalent to the event identified by the location of this descriptor (the source event).

**target_listed:** This 1-bit field signals whether the service defined by the *original_network_id*, *transport_stream_id*, and *service_id* is included in the SDT carried in that TS. When *target_listed* is set to 0b1, the service shall be included in the SDT, otherwise it may not be. In the latter case, the following conditions shall be met:

- the *service_type* for the service shall be 0x19 (H.264/AVC HD digital television service) if the events are simulcast, otherwise the *service_type* shall be the same as for the service where the source event is carried;

- EIT present/following information shall be available for the service in that TS;

- the service shall be running.

**event_simulcast:** This 1-bit field shall be set to 0b1 when the target event and the source event are being simulcast. It shall be set to 0b0 when the events are offset in time.

### 6.2.19.4    Extended event linkage

Several events can be signalled as equivalent using the using the `extended_event_linkage_info` structure in the `linkage_descriptor`. The event being linked to may be a simulcast or may be time offset.

The `extended_event_linkage_info` shall be coded according to table 65.

**Table 65: Extended event linkage info**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `extended_event_linkage_info() {` | | |
| `    loop_length` | 8 | uimsbf |
| `    for (i=0;i<N;i++) {` | | |
| `        target_event_id` | 16 | uimsbf |
| `        target_listed` | 1 | bslbf |
| `        event_simulcast` | 1 | bslbf |
| `        link_type` | 2 | uimsbf |
| `        target_id_type` | 2 | uimsbf |
| `        original_network_id_flag` | 1 | bslbf |
| `        service_id_flag` | 1 | bslbf |
| `        if (target_id_type == 3) {` | | |
| `            user_defined_id` | 16 | uimsbf |
| `        } else {` | | |
| `            if (target_id_type == 1) {` | | |
| `                target_transport_stream_id` | 16 | uimsbf |
| `            }` | | |
| `            if (original_network_id_flag == 0b1) {` | | |
| `                target_original_network_id` | 16 | uimsbf |
| `            }` | | |
| `            if (service_id_flag == 0b1) {` | | |
| `                target_service_id` | 16 | uimsbf |
| `            }` | | |
| `        }` | | |
| `    }` | | |
| `}` | | |

Semantics for the extended event linkage info:

*loop_length*: This 8-bit field indicates the size in bytes of the following loop.

*target_event_id*: This 16-bit field identifies the *event_id* of the event (the target event), carried on the service defined by the *original_network_id*, *transport_stream_id*, and *service_id*, which is equivalent to the event identified by the location of this descriptor (the source event).

*target_listed*: This 1-bit field signals whether the service defined by the *original_network_id*, *transport_stream_id*, and *service_id* is included in the SDT carried in that TS. When *target_listed* is set to 0b1, the service shall be included in the SDT, otherwise it may not be. In the latter case, the following conditions shall be met:

- the *service_type* for the service shall be 0x19 (H.264/AVC HD digital television service) if the events are simulcast, otherwise the *service_type* shall be the same as for the service where the source event is carried;

- EIT present/following information shall be available for the service in that TS;

- the service shall be running.

*event_simulcast*: This 1-bit field shall be set to 0b1 when the target event and the source event are being simulcast. It shall be set to 0b0 when the events are offset in time.

*link_type*: This 2-bit field indicates the type of the target service. It shall be coded according to table 66.

**Table 66: Link type coding**

| link_type | linkage_type (see note) | Type of target service |
|---|---|---|
| 0 | 0x0E | Standard Definition (SD) |
| 1 | 0x0E | High Definition (HD) |
| 2 | 0x0E | frame compatible plano-stereoscopic H.264/AVC |
| 3 | 0x0E | service compatible plano-stereoscopic Multi-View video Coding (MVC) |
| 0 | 0x0F | Ultra High Definition (UHD) |
| 1 | 0x0F | service frame compatible plano-stereoscopic |
| 2 to 3 | 0x0F | reserved for future use |
| 0 to 3 | 0x10 to 0x1F | reserved for future use |
| NOTE: | See table 60. | |

**target_id_type:** This 2-bit field, together with the *original_network_id_flag*, and the *service_id_flag*, identifies the target service or services. A value of zero indicates that the *transport_stream_id* shall be used to identify a single target service. A value of one indicates that the *target_transport_stream_id* shall be used instead of the *transport_stream_id* to identify a single target service. A value of two indicates that the target services can be in one or more DVB transport streams (wildcarded Tsid). A value of 3 indicates the target services are matched using a user defined identifier. The *target_id_type* field shall be coded according to table 67.

**Table 67: Target id type coding**

| target_id_type | How target service is matched |
|---|---|
| 0 | use *transport_stream_id* |
| 1 | use *target_transport_stream_id* |
| 2 | match any *transport_stream_id* (wildcard) |
| 3 | use *user_defined_id* |

Table 68 shows which identifiers shall be used for matching target services, controlled by the combination of the *target_id_type*, the *original_network_id_flag*, and the *service_id_flag* fields.

**Table 68: Target service matching rules**

| target_id_type | original_network_id_flag | service_id_flag | match transport_stream_id ? | match original_network_id ? | match service_id ? | match target_transport_stream_id ? | match target_original_network_id ? | match target_service_id ? |
|---|---|---|---|---|---|---|---|---|
| 0 | 0b0 | 0b0 | ✓ | ✓ | ✓ | | | |
| 0 | 0b0 | 0b1 | ✓ | ✓ | | | | ✓ |
| 0 | 0b1 | 0b0 | ✓ | | ✓ | | ✓ | |
| 0 | 0b1 | 0b1 | ✓ | | | | ✓ | ✓ |
| 1 | 0b0 | 0b0 | | ✓ | ✓ | ✓ | | |
| 1 | 0b0 | 0b1 | | ✓ | | ✓ | | ✓ |
| 1 | 0b1 | 0b0 | | | ✓ | ✓ | ✓ | |
| 1 | 0b1 | 0b1 | | | | ✓ | ✓ | ✓ |
| 2 (see note) | 0b0 | 0b0 | | ✓ | ✓ | | | |
| 2 (see note) | 0b0 | 0b1 | | ✓ | | | | ✓ |
| 2 (see note) | 0b1 | 0b0 | | | ✓ | | ✓ | |
| 2 (see note) | 0b1 | 0b1 | | | | | ✓ | ✓ |
| 3 | n/a | n/a | All services matched with *user_defined_id* | | | | | |
| NOTE: | When *target_id_type* is set to two, neither *transport_stream_id*, nor *target_transport_stream_id* are used for matching. Instead, all services with matching remaining identifiers as shown, are considered matches. | | | | | | | |

*Original_network_id_flag*: This 1-bit flag, which when set to 0b1 indicates that the *target_original_network_id* shall be used instead of the *original_network_id* to determine the target service. See also table 68.

*Service_id_flag*: This 1-bit flag, which when set to 0b1 indicates that the *target_service_id* shall be used instead of the *service_id* to determine the target service. See also table 68.

*User_defined_id*: If this 16-bit field is used, the `linkage_descriptor` should be in the scope of a `private_data_specifier_descriptor`, so the receiver is able to determine the meaning of the *user_defined_id* field.

*Target_transport_stream_id*: This is a 16-bit field which identifies an alternate TS containing the information service indicated, under the control of the *target_id_type*, the *original_network_id_flag*, and the *service_id_flag* fields (see table 68).

*Target_original_network_id*: This 16-bit field gives the label identifying the *network_id* of an alternate originating delivery system of the information service indicated, under the control of the *target_id_type*, the *original_network_id_flag*, and the *service_id_flag* fields (see table 68).

*Target_service_id*: This is a 16-bit field which identifies an alternate information service indicated, under the control of the *target_id_type*, the *original_network_id_flag*, and the *service_id_flag* fields (see table 68).

## 6.2.20    Local time offset descriptor

The `local_time_offset_descriptor` (see table 69) may be used in the TOT to describe country specific dynamic changes of the local time offset relative to UTC.

**Table 69: Local time offset descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| local_time_offset_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       country_code | 24 | bslbf |
|       country_region_id | 6 | bslbf |
|       reserved_future_use | 1 | bslbf |
|       local_time_offset_polarity | 1 | bslbf |
|       local_time_offset | 16 | bslbf |
|       time_of_change | 40 | bslbf |
|       next_time_offset | 16 | bslbf |
|    } | | |
| } | | |

Semantics for the local time offset descriptor:

*country_code*: This 24-bit field identifies a country or group of countries. It shall be coded as three characters, where each character is coded into 8-bits according to ISO/IEC 8859-1 [38] and inserted in order into the field. Individual countries shall be identified by using the alpha-3 code as specified in ISO 3166 [34]. A group of countries shall be identified by the string representation of a decimal number in the range 900 to 999. These numbers shall be coded according to ETSI TS 101 162 [15]. Country codes for groups of countries shall be limited to those within a single time zone.

    EXAMPLE:        United Kingdom has 3-character code "GBR", which is coded as:
               0b0100 0111 0100 0010 0101 0010.

*country_region_id*: This 6-bit field identifies a zone according to table 70, in the country which is indicated by *country_code*. This field is set to 0b00 0000 when there are no different local time zones in the country.

**Table 70: Country region id coding**

| country_region_id | Description |
|---|---|
| 0b00 0000 | no time zone extension used |
| 0b00 0001 | time zone 1 (most easterly region) |
| 0b00 0010 | time zone 2 |
| … | … |
| 0b11 1100 | time zone 60 |
| 0b11 1101 to 0b11 1111 | reserved for future use |

*local_time_offset_polarity*: This 1-bit information indicates the polarity of the following *local_time_offset* and *next_time_offset*. If this bit is set to 0b0 the polarity is positive and the local time is ahead of UTC. If this bit is set to 0b1 the polarity is negative and the local time is behind UTC.

*local_time_offset*: This 16-bit field contains the offset time from UTC at a time when current UTC time is early with respect to *time_of_change*. In conjunction with the *local_time_offset_polarity*, this indicates the time offset in the area which is indicated by the combination of *country_code* and *country_region_id*. These 16 bits are coded as 4 digits in 4-bit BCD in the order hour tens, hour, minute tens, and minutes.

*time_of_change*: This 40-bit field contains the point in time when the time change takes place in UTC and MJD (see annex C). This field is coded as 16 bits giving the 16 lsb of MJD followed by 24 bits coded as 6 digits in 4-bit BCD.

*next_time_offset*: This 16-bit field contains the offset time from UTC at a time when current UTC time is equal to or after *time_of_change*. In conjunction with the *local_time_offset_polarity*, this indicates the time offset in the area which is indicated by the combination of *country_code* and *country_region_id*. These 16 bits are coded as 4 digits in 4-bit BCD in the order hour tens, hour, minute tens and minutes.

## 6.2.21    Mosaic descriptor

A mosaic component is a collection of different video images to form a coded video component. The information is organized so that each specific information when displayed appears on a small area of a screen.

The mosaic_descriptor gives a partitioning of a digital video component into elementary cells, the allocation of elementary cells to logical cells, and gives a link between the content of the logical cell and the corresponding information (e.g. bouquet, service, event, etc.), see table 71.

**Table 71: Mosaic descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| mosaic_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| mosaic_entry_point | 1 | bslbf |
| number_of_horizontal_elementary_cells | 3 | uimsbf |
| reserved_future_use | 1 | bslbf |
| number_of_vertical_elementary_cells | 3 | uimsbf |
| for (i=0;i<N;i++) { | | |
| logical_cell_id | 6 | uimsbf |
| reserved_future_use | 7 | bslbf |
| logical_cell_presentation_info | 3 | uimsbf |
| elementary_cell_field_length | 8 | uimsbf |
| for (j=0;j<N;j++) { | | |
| reserved_future_use | 2 | bslbf |
| elementary_cell_id | 6 | uimsbf |
| } | | |
| cell_linkage_info | 8 | uimsbf |
| if (cell_linkage_info == 0x01) { | | |
| bouquet_id | 16 | uimsbf |
| } | | |
| if (cell_linkage_info == 0x02) { | | |
| original_network_id | 16 | uimsbf |
| transport_stream_id | 16 | uimsbf |
| service_id | 16 | uimsbf |
| } | | |
| if (cell_linkage_info == 0x03) { | | |
| original_network_id | 16 | uimsbf |
| transport_stream_id | 16 | uimsbf |
| service_id | 16 | uimsbf |
| } | | |
| if (cell_linkage_info == 0x04) { | | |
| original_network_id | 16 | uimsbf |
| transport_stream_id | 16 | uimsbf |
| service_id | 16 | uimsbf |
| event_id | 16 | uimsbf |
| } | | |
| } | | |
| } | | |

Semantics for the mosaic descriptor:

*mosaic_entry_point***:** This is a 1-bit field which when set to a value of 0b1 indicates that the mosaic is the highest mosaic in a hierarchy. A complete mosaic system could be organized in a tree structure, the flag being set to identify the entry point in the tree.

*number_of_horizontal_elementary_cells***:** This 3-bit field indicates the number of cells of horizontal screen display. It shall be coded according to table 72.

**Table 72: Number of horizontal elementary cells coding**

| *number_of_horizontal_elementary_cells* | Description |
|---|---|
| 0 | one cell |
| 1 | two cells |
| 2 | three cells |
| 3 | four cells |
| 4 | five cells |
| 5 | six cells |
| 6 | seven cells |
| 7 | eight cells |

*number_of_vertical_elementary_cells***:** This 3-bit field indicates the number of cells of vertical screen display. It shall be coded according to table 73.

**Table 73: Number of vertical elementary cells coding**

| number_of_vertical_elementary_cells | Description |
|---|---|
| 0 | one cell |
| 1 | two cells |
| 2 | three cells |
| 3 | four cells |
| 4 | five cells |
| 5 | six cells |
| 6 | seven cells |
| 7 | eight cells |

*logical_cell_id***:** This 6-bit field is coded in binary form. Different adjacent (see figure 5) elementary cells may be grouped together to form a logical cell. A *logical_cell_id* is associated to such a group of adjacent *elementary_cell_id*. The total number of logical cells shall not exceed the number of elementary cells (maximum = 64). Each elementary cell shall be allocated to one logical cell.

More than one elementary cell may belong to one logical cell.
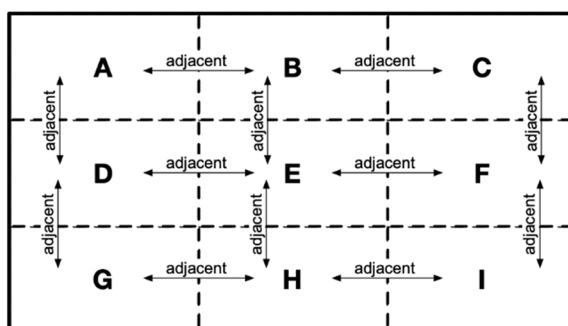


NOTE:       Cells B, D, H, F are adjacent to cell E; C is not adjacent to A or D; D is not adjacent to H.

**Figure 5: Adjacent cells**

*logical_cell_presentation_info***:** This 3-bit field identifies the type of presentation for a logical cell. The *logical_cell_presentation_info* information allows an identification of presentation styles, which are defined in table 74.

**Table 74: Logical cell presentation info coding**

| logical_cell_presentation_info | Description |
|---|---|
| 0 | undefined |
| 1 | video |
| 2 | still picture (see note) |
| 3 | graphics/text |
| 4 to 7 | reserved for future use |
| NOTE:       A coded still picture consists of a video sequence containing exactly one coded picture which is intra-coded. | |

*elementary_cell_field_length***:** The *elementary_cell_field_length* is an 8-bit field specifying the number of bytes following this field up to and including the last *elementary_cell_id* in this *logical_cell_id* loop.

*elementary_cell_id***:** This 6-bit field indicates in binary form the number of the cell. The value of this field is in the range 0 to N=(number_of_horizontal_elementary_cells × number_of_vertical_elementary_cells) − 1.

NOTE:       The elementary cells are implicitly numbered from 0 to N. The value 0 is allocated to the cell of the first row (top left corner). This number is incremented from left to right and from top to bottom in such a way that the number N is allocated to the cell of the last position of the last row (bottom right corner).

*cell_linkage_info***:** This 8-bit field identifies the type of information carried in a logical cell, see table 75 for coding.

**Table 75: Cell linkage info coding**

| cell_linkage_info | Description |
|---|---|
| 0x00 | undefined |
| 0x01 | bouquet related |
| 0x02 | service related |
| 0x03 | other mosaic related |
| 0x04 | event related |
| 0x05 to 0xFF | reserved for future use |

*bouquet_id*: This is a 16-bit field which serves as a label to identify the bouquet described by the cell.

*original_network_id*: This 16-bit field is a label (see clause 5.2) which in conjunction with the following fields uniquely identifies a service, event or mosaic.

*transport_stream_id*: This is a 16-bit field which serves as a label identifying the TS which contains the service, event or mosaic described by the cell.

*service_id*: This is a 16-bit field which serves as a label to identify a service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding `program_map_section`.

The interpretation of this field is context sensitive, dependent on the value of *cell_linkage_info*:

- when *cell_linkage_info* = 0x02, this is the *service_id* of the service described by the cell;

- when *cell_linkage_info* = 0x03, this is the *service_id* of the mosaic service described by the cell;

- when *cell_linkage_info* = 0x04, this is the *service_id* of the service to which the event described by the cell belongs.

*event_id*: This 16-bit field contains the identification number of the described event (uniquely allocated within a service definition).

## 6.2.22    Multilingual bouquet name descriptor

The `multilingual_bouquet_name_descriptor` (see table 76) provides the bouquet name in text form in one or more languages.

**Table 76: Multilingual bouquet name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| multilingual_bouquet_name_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       name_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          char | 8 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the multilingual bouquet name descriptor:

*ISO_639_language_code*: This 24-bit field identifies the language of the following bouquet name. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE:      French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*name_length*: This 8-bit field specifies the length in bytes of the following bouquet name.

*char*: This is an 8-bit field, a sequence of which conveys the name of the bouquet about which the BAT sub_table informs in the language specified. Text information is coded using the character sets and methods described in annex A.

## 6.2.23 Multilingual component descriptor

The `multilingual_component_descriptor` (see table 77) provides a text description of a component in one or more languages. The component is identified by its component tag value.

**Table 77: Multilingual component name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| multilingual_component_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    component_tag | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       text_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          char | 8 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the multilingual component descriptor:

*component_tag*: This 8-bit field has the same value as the *component_tag* field in the `stream_identifier_descriptor` (if present in the PSI `program_map_section`) for the component stream.

*ISO_639_language_code*: This 24-bit field identifies the language of the following text description of the component. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

    EXAMPLE:     French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*text_length*: This 8-bit field specifies the length in bytes of the following text description.

*char*: This is an 8-bit field, a sequence of which conveys a text description of the component stream. Text information is coded using the character sets and methods described in annex A.

## 6.2.24 Multilingual network name descriptor

The `multilingual_network_name_descriptor` (see table 78) provides the network name in text form in one or more languages.

**Table 78: Multilingual network name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| multilingual_network_name_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       name_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          char | 8 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the multilingual network name descriptor:

***ISO_639_language_code*:** This 24-bit field identifies the language of the following bouquet name. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

***name_length*:** This 8-bit field specifies the length in bytes of the following network name.

***char*:** This is an 8-bit field, a sequence of which conveys the name of the network about which the NIT informs in the language specified. Text information is coded using the character sets and methods described in annex A.

## 6.2.25 Multilingual service name descriptor

The `multilingual_service_name_descriptor` (see table 79) provides the names of the service provider and service in text form in one or more languages.

**Table 79: Multilingual service name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| multilingual_service_name_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       service_provider_name_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          char | 8 | uimsbf |
|       } | | |
|       service_name_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          char | 8 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the multilingual service name descriptor:

***ISO_639_language_code*:** This 24-bit field identifies the language of the following bouquet name. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

***service_provider_name_length*:** This 8-bit field specifies the length in bytes of the following service provider name.

***service_name_length*:** This 8-bit field specifies the length in bytes of the following service name.

***char*:** This is an 8-bit field, a sequence of which conveys the name of the service provider or service. Text information is coded using the character sets and methods described in annex A.

## 6.2.26 NVOD reference descriptor

This descriptor, in conjunction with the time shifted service and time shifted event descriptors, provides a mechanism for efficiently describing a number of services which carry the same sequence of events, but with the start times offset from one another. Such a group of time-shifted services is referred to as NVOD, since a user can at any time access near to the start of an event by selecting the appropriate service of the group.

The `NVOD_reference_descriptor` (see table 80) gives a list of the services which together form a NVOD service. Each service is also described in the appropriate SDT sub_table by a `time_shifted_service_descriptor`, see clause 6.2.45. The `time_shifted_service_descriptor` associates a time shifted service with a *reference_service_id*. The *reference_service_id* is the label under which a full description of the NVOD service is given, but the *reference_service_id* does not itself correspond to any *program_number* in the `program_map_section`.

The `time_shifted_event_descriptor` is used in the event information for each time shifted service. Instead of duplicating the full information for each event, the `time_shifted_event_descriptor` points to a *reference_event_id* in the reference service. The full event information is provided in the event information for the reference service.

The services which make up an NVOD service need not all be carried in the same TS. However, a reference service shall be described in the SI in each TS which carries any services of the NVOD service.

**Table 80: NVOD reference descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| NVOD_reference_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       transport_stream_id | 16 | uimsbf |
|       original_network_id | 16 | uimsbf |
|       service_id | 16 | uimsbf |
|    } | | |
| } | | |

Semantics for the NVOD reference descriptor:

*transport_stream_id*: This is a 16-bit field which serves as a label for identification of this TS from any other multiplex within the delivery system.

*original_network_id*: This 16-bit field gives the label identifying the *network_id* of the originating delivery system.

*service_id*: This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding `program_map_section`, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (NVOD reference services) the *service_id* does not have a corresponding *program_number*.

## 6.2.27   Network name descriptor

The `network_name_descriptor` provides the network name in text form (see table 81).

**Table 81: Network name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| network_name_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the network name descriptor:

*char*: This is an 8-bit field, a sequence of which conveys the name of the delivery system about which the NIT informs. Text information is coded using the character sets and methods described in annex A.

## 6.2.28   Parental rating descriptor

This descriptor (see table 82) gives a rating based on age and allows for extensions based on other rating criteria.

**Table 82: Parental rating descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| parental_rating_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       country_code | 24 | bslbf |
|       rating | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the parental rating descriptor:

*country_code*: This 24-bit field identifies a country or group of countries. It shall be coded as three characters, where each character is coded into 8-bits according to ISO/IEC 8859-1 [38] and inserted in order into the field. Individual countries shall be identified by using the alpha-3 code as specified in ISO 3166 [34]. A group of countries shall be identified by the string representation of a decimal number in the range 900 to 999. These numbers shall be coded according to ETSI TS 101 162 [15].

> EXAMPLE 1:　United Kingdom has 3-character code "GBR", which is coded as:
> 0b0100 0111 0100 0010 0101 0010.

*rating*: This 8-bit field is coded according to table 83, giving the recommended minimum age in years of the end user.

**Table 83: Parental rating coding**

| rating | Description |
|---|---|
| 0x00 | undefined |
| 0x01 to 0x0F | minimum age = rating + 3 years |
| 0x10 to 0xFF | defined by the broadcaster |

> EXAMPLE 2:　0x04 implies that end users should be at least 7 years old.

## 6.2.29　Partial TS descriptor

See clause 7.1.2.

## 6.2.30　PDC descriptor

The PDC_descriptor (see table 84) extends the DVB system with the functionalities of Programme Delivery Control (PDC) (ETSI EN 300 231 [28]). The descriptor carries the Programme Identification Label (PIL) as defined in ETSI EN 300 231 [28]. The PIL contains date and time of the first published start time of a certain event.

**Table 84: PDC descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| PDC_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    reserved_future_use | 4 | bslbf |
|    programme_identification_label | 20 | bslbf |
| } | | |

Semantics for the PDC descriptor:

*programme_identification_label*: This 20-bit field gives the PIL (ETSI EN 300 231 [28]). The structure of the PIL is as shown in figure 6 (bit number 1 is the leftmost bit of the string). The values for the day, month, hour (24), and minute are binary coded.

| Programme identification label | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| M | | | | L | M | | | | L | M | | | | L | M | | | | L |
| day | | | | | month | | | | | hour | | | | | minute | | | | |

**Figure 6: Programme identification label**

## 6.2.31    Private data specifier descriptor

This descriptor (see table 85) is used to identify the specifier of any private descriptors or private fields within descriptors.

**Table 85: Private data specifier descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| private_data_specifier_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| private_data_specifier | 32 | uimsbf |
| } | | |

Semantics for the private data specifier descriptor:

*private_data_specifier*: This field shall be coded according to ETSI TS 101 162 [15].

## 6.2.32    Scrambling descriptor

The scrambling_descriptor (see table 86) indicates the selected mode of operation for the scrambling system. It is located in the program_map_section at the program loop level. For further clarifications on the usage of the scrambling_descriptor refer to annex E.

**Table 86: Scrambling descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| scrambling_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| scrambling_mode | 8 | uimsbf |
| } | | |

Semantics for the scrambling descriptor:

*scrambling_mode*: This 8-bit field identifies the selected mode of the scrambling algorithm (see table 87). The technical details of the scrambling algorithm are available only to bona-fide users upon signature of a Non-Disclosure Agreement (NDA) administered by the DVB Common Scrambling Algorithm Custodian.

**Table 87: Scrambling mode coding**

| scrambling_mode | Description |
|---|---|
| 0x00 | reserved for future use |
| 0x01 | this value indicates use of DVB-Common Scrambling Algorithm Version 1 (CSA1). It is the default mode and shall be used when the scrambling_descriptor is not present in the program map section |
| 0x02 | this value indicates use of DVB-Common Scrambling Algorithm Version 2 (CSA2) |
| 0x03 | this value indicates use of DVB-Common Scrambling Algorithm Version 3 (CSA3) |
| 0x04 to 0x0F | reserved for future use |
| 0x10 | this value indicates use of DVB-Common IPTV Software-oriented Scrambling Algorithm (CISSA) version 1 |
| 0x11 to 0x1F | reserved for future use for DVB-CISSA versions |
| 0x20 to 0x6F | reserved for future use |
| 0x70 to 0x7F | Alliance for Telecommunications Industry Solutions (ATIS) defined (see annex J of ATIS 0800006 [i.6]) |
| 0x80 to 0xFE | user defined |
| 0xFF | reserved for future use |

## 6.2.33  Service descriptor

The service_descriptor (see table 88) provides the names of the service provider and the service in text form together with the *service_type*.

Note that the use of the HEVC digital television service type (0x1F) and HEVC digital television service type with HDR and/or a frame rate of 100 Hz, $\frac{120\ 000}{1\ 001}$ Hz, or 120 Hz, or a resolution greater than 3 840 x 2 160 (0x20) listed in table 89 implies additional semantics and is subject to additional rules (see note 4 in table 89 and see note 5 in table 89).

Note that the use of the VVC digital television service type (0x21) listed in table 89 implies additional semantics and is subject to additional rules (see note 6 in table 89).

Note that the use of the AVS3 digital television service type (0x22) listed in table 89 implies additional semantics and is subject to additional rules (see note 7 in table 89).

**Table 88: Service descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    service_type | 8 | uimsbf |
|    service_provider_name_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
|    service_name_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the service descriptor:

*service_type*: This is an 8-bit field specifying the type of the service. The assignment of *service_type* value for a service is described in annex I. It shall be coded according to table 89.

**Table 89: Service type coding**

| service_type | Description |
|---|---|
| 0x00 | reserved for future use |
| 0x01 | digital television service (see note 1) |
| 0x02 | digital radio sound service (see note 2) |
| 0x03 | teletext service |
| 0x04 | NVOD reference service (see note 1) |
| 0x05 | NVOD time-shifted service (see note 1) |
| 0x06 | mosaic service |
| 0x07 | Frequency Modulation (FM) radio service |
| 0x08 | DVB SRM service (ETSI TS 102 770 [22]) |
| 0x09 | reserved for future use |
| 0x0A | advanced codec digital radio sound service |
| 0x0B | H.264/Advanced Video Coding (AVC) mosaic service |
| 0x0C | data broadcast service |
| 0x0D | reserved for Common Interface (CI) usage (EN 50221 [31]) |
| 0x0E | RCS Map (ETSI EN 301 790 [6]) |
| 0x0F | RCS Forward Link Signalling (FLS) (ETSI EN 301 790 [6]) |
| 0x10 | DVB Multimedia Home Platform (MHP) service |
| 0x11 | HD digital television service |
| 0x12 to 0x15 | reserved for future use |
| 0x16 | H.264/AVC SD digital television service |
| 0x17 | H.264/AVC SD NVOD time-shifted service |
| 0x18 | H.264/AVC SD NVOD reference service |
| 0x19 | H.264/AVC HD digital television service |
| 0x1A | H.264/AVC HD NVOD time-shifted service |
| 0x1B | H.264/AVC HD NVOD reference service |
| 0x1C | H.264/AVC frame compatible plano-stereoscopic HD digital television service (see note 3) |
| 0x1D | H.264/AVC frame compatible plano-stereoscopic HD NVOD time-shifted service (see note 3) |
| 0x1E | H.264/AVC frame compatible plano-stereoscopic HD NVOD reference service (see note 3) |
| 0x1F | HEVC digital television service (see note 4) |
| 0x20 | HEVC UHD digital television service (see note 5) with either: <ul><li>a resolution up to 3 840 x 2 160, HDR and/or a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz or 120 Hz,</li><li>or a resolution greater than 3 840 x 2 160, SDR or HDR, with a frame rate up to 60 Hz.</li></ul> |
| 0x21 | VVC digital television service (see note 6) |
| 0x22 | AVS3 digital television service (see note 7) |
| 0x23 to 0x7F | reserved for future use |
| 0x80 to 0xFE | user defined |
| 0xFF | reserved for future use |
| NOTE 1: MPEG-2 SD material should use this type.<br>NOTE 2: MPEG-1 Layer 2 audio material should use this type.<br>NOTE 3: For information on the use of these values, see clause I.2.3 and ETSI TS 101 547-2 [16].<br>NOTE 4: For rules on the use of this value, see clause I.2.5 and ETSI TS 101 547-4 [18]. This value should be used for backward compatible HLG10 HDR services, and/or backward compatible HFR services which are decodable by HEVC_UHDTV_IRD as defined in ETSI TS 101 154 [14], see clause I.2.5.2.<br>NOTE 5: For rules on the use of these values, see clause I.2.6.<br>NOTE 6: For rules on the use of these values, see clause I.2.8.<br>NOTE 7: For rules on the use of these values, see clause I.2.9. | |

*service_provider_name_length*: This 8-bit field specifies the length in bytes of the following service provider name.

*service_name_length*: This 8-bit field specifies the length in bytes of the following service name.

*char*: This is an 8-bit field, a sequence of which conveys the name of the service provider or service. Text information is coded using the character sets and methods described in annex A.

## 6.2.34 Service availability descriptor

This descriptor (see table 90) may be used in the SDT in a terrestrial network. It provides an identification of the cells in which the service is available or not available.

**Table 90: Service availability descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_availability_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    availability_flag | 1 | bslbf |
|    reserved_future_use | 7 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       cell_id | 16 | uimsbf |
|    } | | |
| } | | |

Semantics for the service availability descriptor:

*availability_flag***:** This 1-bit indicator, when set to 0b1, indicates that the service is available on the cells identified by the *cell_id* in the following loop and not available on the other cells. When the bit is set to 0b0, it indicates that the service is unavailable on the cells identified by the *cell_id* in the following loop and available on the other cells.

*cell_id***:** This is a 16-bit field which uniquely identifies a cell in the terrestrial network.

## 6.2.35 Service list descriptor

The service_list_descriptor (see table 91) provides a means of listing the services by *service_id* and *service_type*.

Note that the use of the HEVC digital television service type listed in table 89 implies additional semantics and is subject to additional rules (see also see note 4 in table 89).

**Table 91: Service list descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_list_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       service_id | 16 | uimsbf |
|       service_type | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the service list descriptor:

*service_id***:** This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding program_map_section, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (NVOD reference services) the *service_id* does not have a corresponding *program_number*.

*service_type***:** This is an 8-bit field specifying the type of the service. The assignment of *service_type* value for a service is described in annex I. It shall be coded according to table 89.

## 6.2.36 Service move descriptor

If it is required to move a service from one TS to another, a mechanism is provided which enables an IRD to track the service between TSs by means of a service_move_descriptor. It shall be coded according to table 92.

**Table 92: Service move descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_move_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    new_original_network_id | 16 | uimsbf |
|    new_transport_stream_id | 16 | uimsbf |
|    new_service_id | 16 | uimsbf |
| } | | |

Semantics for the service move descriptor:

*new_original_network_id*: This field contains the *original_network_id* of the TS in which the service is found after the move.

*new_transport_stream_id*: This field contains the *transport_stream_id* of the TS in which the service is found after the move.

*new_service_id*: This field contains the *service_id* of the service after the move.

## 6.2.37    Short event descriptor

The short_event_descriptor provides the name of the event and a short description of the event in text form (table 93).

**Table 93: Short event descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| short_event_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    ISO_639_language_code | 24 | bslbf |
|    name_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       char | 8 | uimsbf |
|    } | | |
|    text_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       text_char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the short event descriptor:

*ISO_639_language_code*: This 24-bit field identifies the language of the following text fields. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

    EXAMPLE:      French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*name_length*: This 8-bit field specifies the length in bytes of the following event name.

*char*: This is an 8-bit field, a sequence of which conveys the event name. Text information is coded using the character sets and methods described in annex A.

*text_length*: This 8-bit field specifies the length in bytes of the following text describing the event.

*text_char*: This is an 8-bit field, a sequence of which conveys the text description for the event. Text information is coded using the character sets and methods described in annex A.

## 6.2.38 Short smoothing buffer descriptor

A `smoothing_buffer_descriptor` is specified in ISO/IEC 13818-1 [1] which enables the bit-rate of a service to be signalled in the PSI.

For use in DVB SI tables, a more compact and efficient descriptor, the `short_smoothing_buffer_descriptor`, is defined here in table 94.

This descriptor may be included in the EIT present/following and EIT schedule tables to signal the bit-rate for each event.

The bit-rate is expressed in terms of a smoothing buffer size and output leak rate.

The presence of the descriptor in the EIT present/following and EIT schedule tables is optional.

The data flows into and from the smoothing buffer are defined as follows:

- bytes of TS packets belonging to the associated service are input to the smoothing buffer at the time defined by equation 2-4 of ISO/IEC 13818-1 [1] (definition of the mathematical byte delivery schedule). The following packets belong to the service:

  - all TS packets of all elementary streams of the service, i.e. all PIDs which are listed as *elementary_PID* in the extended program information part of the PMT section for the service during the time that the event is transmitted;

  - all TS packets of the PID which is identified as the *program_map_PID* for the service in the PAT at the time that the event is transmitted;

  - all TS packets of the PID which is identified as the *PCR_PID* in the PMT section for the service at the time that the event is transmitted;

- all bytes that enter the buffer also exit it.

**Table 94: Short smoothing buffer descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| short_smoothing_buffer_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    sb_size | 2 | uimsbf |
|    sb_leak_rate | 6 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       reserved_future_use | 8 | bslbf |
|    } | | |
| } | | |

Semantics for the short smoothing buffer descriptor:

*sb_size***:** This 2-bit field indicates the size of the smoothing buffer, and is coded according to table 95.

**Table 95: Smoothing buffer size coding**

| sb_size | Buffer size (bytes) |
|---|---|
| 0 | reserved for future use |
| 1 | 1 536 |
| 2 | reserved for future use |
| 3 | reserved for future use |

NOTE: Due to implementation constraints, the specified buffer size value considers spare capacity that may be required in a 2 kbyte Random Access Memory (RAM) for packet jitter.

*sb_leak_rate***:** This 6-bit field indicates the value of the leak rate from the buffer, and is coded according to table 96.

**Table 96: Smoothing buffer leak rate coding**

| sb_leak_rate | Leak rate (Mbit/s) |
|---|---|
| 0 | reserved for future use |
| 1 | 0,0009 |
| 2 | 0,0018 |
| 3 | 0,0036 |
| 4 | 0,0072 |
| 5 | 0,0108 |
| 6 | 0,0144 |
| 7 | 0,0216 |
| 8 | 0,0288 |
| 9 | 0,075 |
| 10 | 0,5 |
| 11 | 0,5625 |
| 12 | 0,8437 |
| 13 | 1,0 |
| 14 | 1,1250 |
| 15 | 1,5 |
| 16 | 1,6875 |
| 17 | 2,0 |
| 18 | 2,2500 |
| 19 | 2,5 |
| 20 | 3,0 |
| 21 | 3,3750 |
| 22 | 3,5 |
| 23 | 4,0 |
| 24 | 4,5 |
| 25 | 5,0 |
| 26 | 5,5 |
| 27 | 6,0 |
| 28 | 6,5 |
| 29 | 6,7500 |
| 30 | 7,0 |
| 31 | 7,5 |
| 32 | 8,0 |
| 33 | 9,0 |
| 34 | 10,0 |
| 35 | 11,0 |
| 36 | 12,0 |
| 37 | 13,0 |
| 38 | 13,5 |
| 39 | 14,0 |
| 40 | 15,0 |
| 41 | 16,0 |
| 42 | 17,0 |
| 43 | 18,0 |
| 44 | 20,0 |
| 45 | 22,0 |
| 46 | 24,0 |
| 47 | 26,0 |
| 48 | 27,0 |
| 49 | 28,0 |
| 50 | 30,0 |
| 51 | 32,0 |
| 52 | 34,0 |
| 53 | 36,0 |
| 54 | 38,0 |
| 55 | 40,0 |
| 56 | 44,0 |
| 57 | 48,0 |
| 58 | 54,0 |
| 59 | 72,0 |

| sb_leak_rate | Leak rate (Mbit/s) |
|---|---|

| sb_leak_rate | Leak rate (Mbit/s) |
|---|---|
| 60 | 108,0 |
| 61 to 63 | reserved for future use |

## 6.2.39     Stream identifier descriptor

The `stream_identifier_descriptor` (see table 97) may be used in the PSI PMT to label component streams of a service so that they can be differentiated, e.g. by text descriptions given in `component_descriptor` in the EIT if present. The `stream_identifier_descriptor` shall be located within the relevant *ES_info* descriptor loop of the PMT.

**Table 97: Stream identifier descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| stream_identifier_descriptor() { | | |
|     descriptor_tag | 8 | uimsbf |
|     descriptor_length | 8 | uimsbf |
|     component_tag | 8 | uimsbf |
| } | | |

Semantics for the stream identifier descriptor:

*component_tag***:** This 8-bit field identifies the component stream for associating it with a description given in a `component_descriptor`. Within a `program_map_section` each `stream_identifier_descriptor` shall have a different value for this field.

## 6.2.40     Stuffing descriptor

The `stuffing_descriptor` provides a means of invalidating previously coded descriptors or inserting dummy descriptors for table stuffing (see table 98).

**Table 98: Stuffing descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| stuffing_descriptor() { | | |
|     descriptor_tag | 8 | uimsbf |
|     descriptor_length | 8 | uimsbf |
|     for (i=0;i<N;i++) { | | |
|         stuffing_byte | 8 | bslbf |
|     } | | |
| } | | |

Semantics for the stuffing descriptor:

*stuffing_byte***:** This is an 8-bit field. Each occurrence of the field may be set to any value. The IRD may discard the stuffing bytes.

## 6.2.41     Subtitling descriptor

In the ISO/IEC 13818-1 [1] PMT the value of *stream_type* for any PID carrying DVB subtitles as defined by ETSI EN 300 743 [2] shall be 0x06 (this indicates a Packetized Elementary Stream (PES) carrying private data). See table 99.

**Table 99: Subtitling descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| subtitling_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       subtitling_type | 8 | bslbf |
|       composition_page_id | 16 | bslbf |
|       ancillary_page_id | 16 | bslbf |
|    } | | |
| } | | |

Semantics for the subtitling descriptor:

***ISO_639_language_code*:** This 24-bit field identifies the language of the subtitle. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

    EXAMPLE:    French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

***subtitling_type*:** This 8-bit field provides information on the content of the subtitle and the intended display. The coding of this field shall use the codes in the range 0x10 to 0x2F defined for *component_type* when *stream_content* is 0x03 in table 26.

***composition_page_id*:** This 16-bit field identifies the composition page. DVB_subtitling_segment signalling this *page_id* shall be decoded if the previous data in the subtitling_descriptor matches the user's selection criteria.

    NOTE 1:  The *composition_page_id* is signalled in at least the DVB_subtitling_segment that define the data structure of the subtitle screen; the page_composition_segment and region_composition_segment. It may additionally be signalled in segments containing data on which the composition depends.

***ancillary_page_id*:** This identifies the (optional) ancillary page. DVB_subtitling_segment signalling this *page_id* shall also be decoded if the previous data in the subtitling_descriptor matches the user's selection criteria. The values in the *ancillary_page_id* and the *composition_page_id* fields shall be the same if no ancillary page is provided.

    NOTE 2:  The *ancillary_page_id* is never signalled in a composition segment. It may be signalled in Colour Look-Up Table (CLUT) definition segments, object segments and any other type of segment.

    NOTE 3:  (Terminology): A segment that signals a particular page number in its *page_id* field is said to be "in" that page. The page is said to "contain" that segment.

## 6.2.42    Telephone descriptor

The telephone_descriptor may be used to indicate a telephone number which may be used in conjunction with a modem (Public Switched Telephone Network (PSTN) or cable) to exploit narrowband interactive channels.

The telephone_descriptor syntax is specified in table 100.

**Table 100: Telephone descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `telephone_descriptor() {` | | |
| `    descriptor_tag` | 8 | uimsbf |
| `    descriptor_length` | 8 | uimsbf |
| `    reserved_future_use` | 2 | bslbf |
| `    foreign_availability` | 1 | bslbf |
| `    connection_type` | 5 | uimsbf |
| `    reserved for future use` | 1 | bslbf |
| `    country_prefix_length` | 2 | uimsbf |
| `    international_area_code_length` | 3 | uimsbf |
| `    operator_code_length` | 2 | uimsbf |
| `    reserved for future use` | 1 | bslbf |
| `    national_area_code_length` | 3 | uimsbf |
| `    core_number_length` | 4 | uimsbf |
| `    for (i=0;i<N;i++) {` | | |
| `        country_prefix_char` | 8 | uimsbf |
| `    }` | | |
| `    for (i=0;i<N;i++) {` | | |
| `        international_area_code_char` | 8 | uimsbf |
| `    }` | | |
| `    for (i=0;i<N;i++) {` | | |
| `        operator_code_char` | 8 | uimsbf |
| `    }` | | |
| `    for (i=0;i<N;i++) {` | | |
| `        national_area_code_char` | 8 | uimsbf |
| `    }` | | |
| `    for (i=0;i<N;i++) {` | | |
| `        core_number_char` | 8 | uimsbf |
| `    }` | | |
| `}` | | |

Semantics for the telephone descriptor:

*foreign_availability*: This is a 1-bit flag. When set to 0b1 it indicates that it is possible to call the number from outside of the country specified by the country prefix. When set to 0b0 it indicates that it is possible to call the number from inside the country specified by the country prefix only.

*connection_type*: This is a 5-bit field which indicates connection types. The meaning of different *connection_type* values is outside the scope of the present document.

> EXAMPLE:     One example of the use of the *connection_type* is to inform the IRD that when, if an interaction is initiated, if the connection is not made within 1 minute, then the connection attempt is to be aborted.

*country_prefix_length*: This 2-bit field specifies the number of 8-bit alphanumeric characters in the country prefix.

*international_area_code_length*: This 3-bit field specifies the number of 8-bit alphanumeric characters in the international area code.

*operator_code_length*: This 2-bit field specifies the number of 8-bit alphanumeric characters in the operator code.

*national_area_code_length*: This 3-bit field specifies the number of 8-bit alphanumeric characters in the national area code.

*core_number_length*: This 4-bit field specifies the number of 8-bit alphanumeric characters in the core number.

*country_prefix_char*: This is an 8-bit field, a sequence of which conveys the country prefix. It shall be coded in accordance with ISO/IEC 8859-1 [38].

*international_area_code_char*: This is an 8-bit field, a sequence of which conveys the international area code. It shall be coded in accordance with ISO/IEC 8859-1 [38].

*operator_code_char*: This is an 8-bit field, a sequence of which conveys the operator code. It shall be coded in accordance with ISO/IEC 8859-1 [38].

*national_area_code_char***:** This is an 8-bit field, a sequence of which conveys the national area code. It shall be coded in accordance with ISO/IEC 8859-1 [38].

*core_number_char***:** This is an 8-bit field, a sequence of which conveys the core number. It shall be coded in accordance with ISO/IEC 8859-1 [38].

## 6.2.43   Teletext descriptor

The `teletext_descriptor` (see table 101) shall be used in the PSI PMT to identify streams which carry EBU teletext data. The descriptor shall be located within the relevant *ES_info* descriptor loop of the PMT.

**Table 101: Teletext descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| teletext_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       teletext_type | 5 | uimsbf |
|       teletext_magazine_number | 3 | uimsbf |
|       teletext_page_number | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the teletext descriptor:

*ISO_639_language_code***:** This 24-bit field identifies the language of the teletext. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

> EXAMPLE:       French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*teletext_type***:** This 5-bit field indicates the type of teletext page indicated. It shall be coded according to table 102.

**Table 102: Teletext type coding**

| teletext_type | Description |
|---|---|
| 0x00 | reserved for future use |
| 0x01 | initial teletext page |
| 0x02 | teletext subtitle page |
| 0x03 | additional information page |
| 0x04 | programme schedule page |
| 0x05 | teletext subtitle page for hearing impaired people |
| 0x06 to 0x1F | reserved for future use |

*teletext_magazine_number***:** This is a 3-bit field which identifies the magazine number as defined in ETSI EN 300 706 [30].

*teletext_page_number***:** This is an 8-bit field giving two 4-bit hex digits identifying the page number as defined in ETSI EN 300 706 [30].

## 6.2.44   Time shifted event descriptor

The `time_shifted_event_descriptor` (see table 103) is used in place of the `short_event_descriptor` to indicate an event which is a time-shifted copy of another event.

**Table 103: Time shifted event descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| time_shifted_event_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    reference_service_id | 16 | uimsbf |
|    reference_event_id | 16 | uimsbf |
| } | | |

Semantics for the time shifted event descriptor:

***reference_service_id*:** This 16-bit field identifies the reference service of a NVOD collection of services. The reference service can always be found in this DVB transport stream. The *service_id* here does not have a corresponding *program_number* in the program_map_section.

***reference_event_id*:** This 16-bit field identifies the reference event of which the event described by this descriptor is a time-shifted copy.

## 6.2.45 Time shifted service descriptor

This descriptor is used in place of the service_descriptor to indicate services which are time shifted copies of other services (see table 104).

**Table 104: Time shifted service descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| time_shifted_service_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    reference_service_id | 16 | uimsbf |
| } | | |

Semantics for the time shifted service descriptor:

***reference_service_id*:** This 16-bit field identifies the reference service of a NVOD collection of services. The reference service can always be found in this DVB transport stream. The *service_id* here does not have a corresponding *program_number* in the program_map_section.

## 6.2.46 Transport stream descriptor

The transport_stream_descriptor (see table 105), being transmitted in the TSDT (see ISO/IEC 13818-1 [1]) only, may be used to indicate the compliance of a transport stream with an Moving Pictures Expert Group (MPEG) based system, e.g. DVB.

**Table 105: Transport stream descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| transport_stream_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the transport stream descriptor:

***byte*:** This is an 8-bit field. For identification of DVB transport streams the *descriptor_length* field shall be set to the value 0x03 indicating three following bytes. The three bytes shall contain the values 0x44, 0x56, and 0x42 (ASCII encoding of the string "DVB").

## 6.2.47  VBI data descriptor

The `VBI_data_descriptor` (see table 106) shall be used in the PSI PMT of a stream which carries VBI data as defined in ETSI EN 301 775 [5]. The appropriate *ES_info* field of the `program_map_section` describing a VBI data stream shall contain one and only one `VBI_data_descriptor`.

For transmission of multiple VBI data streams in one service a `VBI_data_descriptor` in each of the component loops of the `program_map_section` is required.

**Table 106: VBI data descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `VBI_data_descriptor() {` | | |
| `    descriptor_tag` | 8 | uimsbf |
| `    descriptor_length` | 8 | uimsbf |
| `    for (i=0;i<N;i++) {` | | |
| `        data_service_id` | 8 | uimsbf |
| `        data_service_descriptor_length` | 8 | uimsbf |
| `        if (data_service_id == 0x01` | | |
| `            || data_service_id == 0x02` | | |
| `            || data_service_id == 0x04` | | |
| `            || data_service_id == 0x05` | | |
| `            || data_service_id == 0x06` | | |
| `            || data_service_id == 0x07) {` | | |
| `        for (j=0;j<N;j++) {` | | |
| `            reserved_future_use` | 2 | bslbf |
| `            field_parity` | 1 | bslbf |
| `            line_offset` | 5 | uimsbf |
| `        }` | | |
| `        } else {` | | |
| `        for (j=0;j<N;j++) {` | | |
| `            reserved_future_use` | 8 | bslbf |
| `        }` | | |
| `    }` | | |
| `    }` | | |
| `}` | | |

Semantics for the VBI data descriptor:

*data_service_id***:** This 8-bit field identifies a VBI service type contained in the elementary stream to which the `VBI_data_descriptor` is associated. Its value is defined in table 107. For every VBI service contained in the VBI data stream there shall be one and only one *data_service_id* coded in the `VBI_data_descriptor`. However, a VBI service may temporarily not be present in the VBI data stream while its *data_service_id* still is present in the `VBI_data_descriptor`. This discrepancy may last no longer than 10 s.

**Table 107: Data service id coding**

| *data_service_id* | Description |
|---|---|
| 0x00 | reserved for future use |
| 0x01 | EBU teletext (requires additional `teletext_descriptor`) |
| 0x02 | inverted teletext |
| 0x03 | reserved for future use |
| 0x04 | Video Programme System (VPS) |
| 0x05 | Wide Screen Signalling (WSS) |
| 0x06 | closed captioning |
| 0x07 | monochrome 4:2:2 samples |
| 0x08 to 0xEF | reserved for future use |
| 0xF0 to 0xFF | user defined |

*data_service_descriptor_length***:** This 8-bit field counts the number of bytes immediately following this field used for the description of the service indicated by *data_service_id*.

The descriptor itself contains one entry (byte) for each VBI line on which the coded data of the associated stream is intended to be presented.

*field_parity***:** This 1-bit flag specifies the field for which the associated coded data is intended to be presented if it is transcoded into the VBI. The value 0b1 indicates the first (odd) field of a frame, the value 0b0 indicates the second (even) field of a frame. Within a data service descriptor, first all descriptor entries concerning the first field shall be given (if any), followed by all descriptor entries concerning the second field (if any).

*line_offset***:** This 5-bit field specifies the line number on which the associated coded data is intended to be presented if it is transcoded into the VBI. Within a field, the *line_offset* numbering shall follow a progressive incremental order. The *line_offset* parameter follows the definition of the associated data field type (see clauses 2.4 to 2.8 in ETSI EN 301 775 [5]). A *line_offset* referred to as "line number undefined" shall not be used in the `VBI_data_descriptor`.

The encoder shall ensure that at least all coded lines in the associated elementary stream data have their counterpart coded in the `VBI_data_descriptor` (and therefore in the appropriate data service descriptor). A specific *line_offset* and *field_parity* combination shall not be coded more than once in a single `VBI_data_descriptor`.

## 6.2.48    VBI teletext descriptor

The `VBI_teletext_descriptor` (see table 108) shall be used in the PSI PMT to identify streams which carry VBI data as well as EBU teletext data. The descriptor shall be located within the relevant *ES_info* descriptor loop of the PMT.

**Table 108: VBI teletext descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| VBI_teletext_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ISO_639_language_code | 24 | bslbf |
|       teletext_type | 5 | uimsbf |
|       teletext_magazine_number | 3 | uimsbf |
|       teletext_page_number | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the VBI teletext descriptor:

The semantics for the `VBI_teletext_descriptor` is the same as defined for the `teletext_descriptor` in clause 6.2.43. The only exception is that the `VBI_teletext_descriptor` is not to be used to associate *stream_type* 0x06 with the VBI standard nor the EBU teletext standard. Decoders can only use the languages in this descriptor to select magazines and subtitles.

# 6.3    Extended descriptor identification and location

All extended descriptors are based on the `extension_descriptor` (see clause 6.2.16).

Table 109 lists the extended descriptors declared or defined within the present document, giving the *descriptor_tag_extension* values and the intended placement within the SI tables. This does not imply that their use in other tables is restricted.

Table 109 uses the following mark-up to indicate the possible locations of descriptors:

- a check mark ("✓") indicates that the descriptor may be carried in the respective table;

- a dash ("-") indicates that the descriptor shall not be carried in the respective table;

- an empty table cell indicates that nothing is implied regarding the carriage of the descriptor in the respective table.

**Table 109: Possible locations of extended descriptors**

| Extended descriptor | Tag extension value | NIT | BAT | SDT | EIT | TOT | PMT | SIT (see note) |
|---|---|---|---|---|---|---|---|---|
| image_icon_descriptor | 0x00 | ✓ | ✓ | ✓ | ✓ | - | - | ✓ |
| cpcm_delivery_signalling_descriptor (see ETSI TS 102 825 [27] and ETSI TR 102 825 [i.5]) | 0x01 | - | - | ✓ | ✓ | - | - | - |
| CP_descriptor (see ETSI TS 102 825 [27] and ETSI TR 102 825 [i.5]) | 0x02 | - | - | - | - | - | ✓ | - |
| CP_identifier_descriptor (see ETSI TS 102 825 [27] and ETSI TR 102 825 [i.5]) | 0x03 | ✓ | ✓ | ✓ | ✓ | - | - | - |
| T2_delivery_system_descriptor | 0x04 | ✓ | - | - | - | - | - | - |
| SH_delivery_system_descriptor | 0x05 | ✓ | - | - | - | - | - | - |
| supplementary_audio_descriptor | 0x06 | - | - | - | - | - | ✓ | - |
| network_change_notify_descriptor | 0x07 | ✓ | - | - | - | - | - | - |
| message_descriptor | 0x08 | ✓ | ✓ | ✓ | ✓ | - | - | - |
| target_region_descriptor | 0x09 | ✓ | ✓ | ✓ | - | - | - | - |
| target_region_name_descriptor | 0x0A | ✓ | ✓ | - | - | - | - | - |
| service_relocated_descriptor | 0x0B | - | - | ✓ | - | - | - | - |
| XAIT_PID_descriptor (see ETSI TS 102 727 [i.2]) | 0x0C | ✓ | ✓ | - | - | - | - | - |
| C2_delivery_system_descriptor | 0x0D | ✓ | - | - | - | - | - | - |
| DTS-HD_descriptor (see annex G) | 0x0E | - | - | - | - | - | ✓ | - |
| DTS_Neural_descriptor (see annex L) | 0x0F | - | - | - | - | - | ✓ | - |
| video_depth_range_descriptor | 0x10 | - | - | ✓ | ✓ | - | - | - |
| T2MI_descriptor | 0x11 | - | - | - | - | - | ✓ | - |
| reserved for future use | 0x12 | | | | | | | |
| URI_linkage_descriptor | 0x13 | ✓ | ✓ | ✓ | ✓ | - | ✓ | ✓ |
| CI_ancillary_data_descriptor (see ETSI TS 103 205 [i.3]) | 0x14 | ✓ | ✓ | ✓ | ✓ | - | - | - |
| AC-4_descriptor (see annex D) | 0x15 | - | - | - | - | - | ✓ | - |
| C2_bundle_delivery_system_descriptor | 0x16 | ✓ | - | - | - | - | - | - |
| S2X_satellite_delivery_system_descriptor | 0x17 | ✓ | - | - | - | - | - | - |
| protection_message_descriptor (see ETSI TS 102 809 [25]) | 0x18 | - | - | - | - | - | ✓ | - |
| audio_preselection_descriptor | 0x19 | - | - | - | - | - | ✓ | - |
| reserved for future use | 0x1A to 0x1F | | | | | | | |
| TTML_subtitling_descriptor (see ETSI EN 303 560 [12]) | 0x20 | - | - | - | - | - | ✓ | - |
| DTS-UHD_descriptor (see annex G) | 0x21 | - | - | - | - | - | ✓ | - |
| service_prominence_descriptor | 0x22 | ✓ | ✓ | ✓ | - | - | - | - |
| vvc_subpictures_descriptor | 0x23 | - | - | ✓ | ✓ | - | ✓ | - |
| S2Xv2_satellite_delivery_system_descriptor | 0x24 | ✓ | - | - | - | - | - | - |
| reserved for future use | 0x25 to 0x7F | | | | | | | |
| user defined | 0x80 to 0xFF | | | | | | | |
| NOTE: Only found in Partial Transport Streams. | | | | | | | | |

# 6.4          Extended descriptor coding

## 6.4.0          General principles

When an extension_descriptor (see clause 6.2.16) is used as one of the descriptors in the construct descriptor(), which appears in the sections of clause 5.2, the selector field of this extension_descriptor shall be coded according to clause 6.4.

The following semantics apply to all the descriptors defined in clause 6.4.

*descriptor_tag*: This 8-bit field shall be set to the *descriptor_tag* value assigned for the extension_descriptor in table 12.

*descriptor_length*: The *descriptor_length* is an 8-bit field specifying the total number of bytes of the data portion of the descriptor following the byte defining the value of this field.

***descriptor_tag_extension*:** The *descriptor_tag_extension* is an 8-bit field which identifies the syntax and semantics of the selector field of this `extension_descriptor`. The values of *descriptor_tag_extension* are defined in table 109.

The bit and transmission ordering rules defined in clause 5.1.6 shall apply.

## 6.4.1    Audio preselection descriptor

The `audio_preselection_descriptor` (see table 110) provides information about the available audio preselections for one audio programme delivered by one NGA stream (single-stream delivery) or more than one NGA stream (multi-stream delivery). Information from this, and other descriptors, allows the receiver to select and present the appropriate NGA streams to the user.

In case of single-stream delivery, use of the `audio_preselection_descriptor` is optional.

The descriptor also provides information to enable linking of multiple NGA streams that contain audio programme components of a single audio programme. In this case of multi-stream delivery, only one `audio_preselection_descriptor` shall be present for that audio programme, and it shall be carried in the *ES_info* descriptor loop of the PSI PMT for the main NGA stream and shall contain all audio preselections that require auxiliary NGA streams. *ES_info* descriptor loops of PSI PMT for auxiliary NGA streams which are part of a multi-stream delivery shall include a `stream_identifier_descriptor` which conveys a *component_tag* value matching one of those listed in the `audio_preselection_descriptor` for the main NGA stream.

The `audio_preselection_descriptor` is intended be used in the PMT for stream selection and audio preselection selection in the IRD.

The `audio_preselection_descriptor` is used in the PMT to indicate the main NGA stream of an audio programme. The descriptor also indirectly identifies all auxiliary NGA streams (if present) via the *component_tag* field.

The descriptor should list all audio preselections that the broadcaster intends for automatic selection by IRDs. The descriptor need not list all audio preselections available in an NGA stream. It shall list all audio preselections which contain audio programme components delivered via multiple NGA streams.

If NGA streams are transmitted together with non-NGA streams for the same service it is strongly recommended to include in the `audio_preselection_descriptor` all audio preselections intended for IRD automatic selection. audio preselections which are not included in the `audio_preselection_descriptor` may not be considered by an IRD for automatic selection, and hence may not be offered to the listener. This is because IRDs are not expected to extract preselection information from the elementary stream when performing an automatic selection.

For further rules on using this descriptor in the PMT, see clause M.2.

**Table 110: Audio preselection descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| audio_preselection_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| descriptor_tag_extension | 8 | uimsbf |
| num_preselections | 5 | uimsbf |
| reserved_zero_future_use | 3 | bslbf |
| for (i=0;i<N;i++) { | | |
| preselection_id | 5 | uimsbf |
| audio_rendering_indication | 3 | uimsbf |
| audio_description | 1 | bslbf |
| spoken_subtitles | 1 | bslbf |
| dialogue_enhancement | 1 | bslbf |
| interactivity_enabled | 1 | bslbf |
| language_code_present | 1 | bslbf |
| text_label_present | 1 | bslbf |
| multi_stream_info_present | 1 | bslbf |
| future_extension | 1 | bslbf |
| if (language_code_present == 0b1) { | | |
| ISO_639_language_code | 24 | bslbf |
| } | | |
| if (text_label_present == 0b1) { | | |
| message_id | 8 | uimsbf |
| } | | |
| if (multi_stream_info_present == 0b1) { | | |
| num_aux_components | 3 | uimsbf |
| reserved_zero_future_use | 5 | bslbf |
| for (j=0;j<N;j++) { | | |
| component_tag | 8 | uimsbf |
| } | | |
| } | | |
| if (future_extension == 0b1) { | | |
| reserved_zero_future_use | 3 | bslbf |
| future_extension_length | 5 | uimsbf |
| for (j=0;j<N;j++) { | | |
| future_extension_byte | 8 | uimsbf |
| } | | |
| } | | |
| } | | |
| } | | |

Semantics for the audio preselection descriptor:

*num_preselections*: This 5-bit field indicates the number of audio preselections, that are described in this descriptor. The minimum number of *num_preselections* shall be 1.

*preselection_id*: This 5-bit field identifies this audio preselection. The value of this field shall match the corresponding field in the elementary stream as described in clause M.3. All the audio programme components corresponding to the first audio preselection in the loop are contained in the main NGA stream.

NOTE 1: It may be necessary to signal several audio preselections which differ only by the presence of additional features (e.g. accessibility, interactivity), facilitating automatic selection by receivers (e.g. based on user preferences).The audio_preselection_descriptor may contain a subset of the audio preselections available in the audio elementary stream.

*audio_rendering_indication*: This 3-bit field contains a hint for a preferred reproduction channel layout. It shall be coded according to table 111.

**Table 111: Audio rendering indication coding**

| audio_rendering_indication | Description |
|---|---|
| 0 | no preference given for the reproduction channel layout |
| 1 | preferred reproduction channel layout is stereo |
| 2 | preferred reproduction channel layout is two-dimensional (e.g. 5.1 multi-channel) |
| 3 | preferred reproduction channel layout is three-dimensional |
| 4 | content is pre-rendered for consumption with headphones |
| 5 to 7 | reserved for future use |

**audio_description:** This 1-bit field indicates, when set to 0b1 that this audio preselection contains audio description for the visually impaired (i.e. a spoken description of the visual content of the service). When set to 0b0, it indicates that the audio preselection does not contain audio description for the visually impaired.

**spoken_subtitles:** This 1-bit field indicates, when set to 0b1 that this audio preselection contains spoken subtitles. When set to 0b0, it indicates that the audio preselection does not contain spoken subtitles.

**dialogue_enhancement:** This 1-bit field indicates, when set to 0b1 that this audio preselection provides support for dialogue enhancement. When set to 0b0, it indicates that the audio preselection does not provide support for dialogue enhancement.

> NOTE 2: The dialogue enhancement feature is described in ETSI TS 101 154 [14]. The dialogue enhancement feature of an NGA preselection can be used in a flexible way to enable delivery of a wide variety of enhancements, including clean audio for the hearing impaired.

**interactivity_enabled:** This 1-bit field indicates, when set to 0b1 that the audio preselection provides additional user interactivity. When set to 0b0, it indicates that no additional user interactivity is available.

> NOTE 3: This flag may be used to determine the need for initializing a user interface for audio interactivity.

**language_code_present:** This 1-bit field indicates, when set to 0b1 that an *ISO_639_language_code* field for this audio preselection is present. When set to 0b0, it indicates that no *ISO_639_language_code* field for this audio preselection is present.

**text_label_present:** This 1-bit field indicates, when set to 0b1 that a *message_id* field is present. The `message_descriptor` containing the same *message_id* provides a text label for this audio preselection. When set to 0b0, it indicates that there is no text label for this audio preselection.

**multi_stream_info_present:** This 1-bit field indicates, when set to 0b1 that the audio programme is conveyed over multiple elementary streams within the DVB transport stream. When set to 0b0, it indicates that this PID conveys a complete audio programme. It shall be set to zero for the first audio preselection in the loop.

**future_extension:** This 1-bit flag indicates, when set to 0b1 that the reserved_zero_future_use and the future extension fields are present for this audio preselection. When set to 0b0, it indicates that the reserved_zero_future_use and the future extension fields are not present for this audio preselection.

**ISO_639_language_code:** This 24-bit field identifies the language of the audio preselection. If the audio preselection contains multiple languages, the language code shall be set to the primary language of the audio preselection. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

> EXAMPLE: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

**message_id:** This 8-bit field where present provides the link to a textual description for this audio preselection. The text description is conveyed using a `message_descriptor` with a matching *message_id* within the SDT of the service. Multiple `message_descriptor` with a matching *message_id* may be present, if the text description is available in multiple languages.

> NOTE 4: This text label is intended only for user interface display to aid the end user in choosing a specific preselection, and not for any automated actions by the receiver.

*num_aux_components*: This 3-bit field indicates the number of auxiliary elementary streams (in addition to the main elementary stream) over which the audio data of the audio service is distributed. The value of this field shall not be equal to 0.

*component_tag*: This 8-bit field has the same value as the *component_tag* field in the `stream_identifier_descriptor` (if present in the PSI program map section) for the auxiliary NGA stream.

*future_extension_length*: This 5-bit field gives the total length in bytes of the following loop containing the future extension field.

*future_extension_byte*: This is an 8-bit field. The sequence of *future_extension_byte* fields specifies the future extension field. These bytes are reserved for future extensions of the `audio_preselection_descriptor`.

## 6.4.2    CID ancillary data descriptor

The `CI_ancillary_data_descriptor` (see table 112) is used to convey ancillary data used in the construction of Content Identifier (CID) in companion screen applications, as are defined in ETSI TS 103 286-2 [i.4].

**Table 112: CID ancillary data descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| CI_ancillary_data_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       ancillary_data_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the CID ancillary data descriptor:

*ancillary_data_byte*: This is an 8-bit field. The sequence of *ancillary_data_byte* fields specifies the ancillary data field. The syntax and semantics of the ancillary data field are defined by the companion screen applications specification ETSI TS 103 286-2 [i.4].

## 6.4.3    CP descriptor

The `CP_descriptor` (see table 113) is used to specify both system-wide and specific Content Protection (CP) management information. When the `CP_descriptor` is found in the PMT, the *CP_PID* points to packets containing program related CP information such as SRM and content licence related information.

**Table 113: CP descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| CP_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    CP_system_id | 16 | uimsbf |
|    reserved_future_use | 3 | bslbf |
|    CP_PID | 13 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       private_data_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the CP descriptor:

*CP_system_id*: This 16-bit field indicates the type of CP system applicable. It shall be coded according to ETSI TS 101 162 [15].

*CP_PID***:** This 13-bit field indicates the PID of the DVB transport stream packets which shall contain information for the CP systems as specified with the associated *CP_system_id*.

*private_data_byte***:** This is an 8-bit field, the value of which is privately defined.

## 6.4.4 CP identifier descriptor

The `CP_identifier_descriptor` (see table 114) indicates whether a particular bouquet, service or event is associated with a CP system or carries information relating to a CP system (e.g. CP system metadata or CP system renewability messages). It identifies the CP system and the type of information by means of the *CP_system_id*.

**Table 114: CP identifier descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| CP_identifier_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       CP_system_id | 16 | uimsbf |
|    } | | |
| } | | |

Semantics for the CP identifier descriptor:

*CP_system_id***:** This 16-bit field identifies the CP system and the type of information (e.g. content licence, content metadata, SRM, etc.). It shall be coded according to ETSI TS 101 162 [15].

## 6.4.5 CPCM delivery signalling descriptor

The `cpcm_delivery_signalling_descriptor` conveys Usage State Information (USI) for CPCM systems. Its syntax and semantics are fully defined and described in part 9 of ETSI TS 102 825 [27].

## 6.4.6 Delivery system descriptors

### 6.4.6.1 C2 delivery system descriptor

The `C2_delivery_system_descriptor` (see table 115) shall be used in the TS loop of the NIT to describe DVB Cable Framing and Modulation, Second Generation (DVB-C2) transmissions according to ETSI EN 302 769 [9]. This descriptor maps DVB transport streams to data Physical Layer Pipe (PLP)s in data slices of DVB-C2 systems. The number of `C2_delivery_system_descriptor` per NIT shall equal the number of DVB transport streams in the network carried over DVB-C2.

Each TS loop of the NIT describing a TS which is conveyed within a DVB-C2 system, shall contain either a `C2_delivery_system_descriptor`, or one or more `C2_bundle_delivery_system_descriptor` (see clause 6.4.6.4).

**Table 115: C2 delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| C2_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    plp_id | 8 | uimsbf |
|    data_slice_id | 8 | uimsbf |
|    C2_System_tuning_frequency | 32 | bslbf |
|    C2_System_tuning_frequency_type | 2 | uimsbf |
|    active_OFDM_symbol_duration | 3 | bslbf |
|    guard_interval | 3 | bslbf |
| } | | |

Semantics for the C2 delivery system descriptor:

*plp_id***:** This 8-bit field uniquely identifies a data PLP within the DVB-C2 system. The term is defined in ETSI EN 302 769 [9].

*data_slice_id***:** This 8-bit field uniquely identifies a data slice within the DVB-C2 system. The term is defined in ETSI EN 302 769 [9].

*C2_System_tuning_frequency***:** This 32-bit field indicates the frequency value in steps of 1 Hz that the receiver should tune to. The coding range is from minimum 0x0000 0001 (1 Hz) up to a maximum of 0xFFFF FFFF (4 294 967 295 Hz). This tuning frequency follows the syntax described in the *C2_System_tuning_frequency_type* field.

*C2_System_tuning_frequency_type***:** This 2-bit field indicates the tuning frequency type (see table 116). This differentiation is needed to address all specifics of the DVB-C2 transmit signal and the underlying Data Slice structure.

**Table 116: C2 tuning frequency type coding**

| *C2_System_tuning_frequency_type* | Description |
|---|---|
| 0b00 | Data Slice tuning frequency<br><br>This is the default option for DVB-C2 systems. The *C2_System_tuning_frequency* field conveys the tuning frequency of the data slice to which *plp_id* refers. The *C2_System_tuning_frequency* for a particular Data Slice is the sum of the L1 signalling parameters *START_FREQUENCY* and the *DSLICE_TUNE_POS*. Note that the Data Slice tuning frequency information in the Layer One (first or bottom-most layer) (L1) signalling as well as in the `C2_delivery_system_descriptor` have to be updated synchronously. |
| 0b01 | C2 system centre frequency<br><br>This option is used by DVB-C2 head-ends that are not able to update the Data Slice tuning frequency information in the `C2_delivery_system_descriptor` and the L1 signalling in a synchronous way. The *C2_System_tuning_frequency* is the centre frequency of the DVB-C2 system, and it is required that a complete Preamble can be received. The receiver needs to evaluate the L1 signalling in the preamble to get knowledge of the final tuning position. |
| 0b10 | Initial tuning position for a (dependent) Static Data Slice<br><br>Signalling of this option implies that the Data Slice to be demodulated is a (dependent) Static Slice. In the case of tuning to a (dependent) Static Data Slice, it cannot be guaranteed that the receiver is able to decode the L1 signalling at its final tuning position. Therefore the receiver will first tune to the signalled initial *C2_System_tuning_frequency* where a complete Preamble is transmitted. This frequency will usually be the DVB-C2 system centre frequency, but can be any tuning position where the receiver can reliably decode the L1 signal. The receiver needs to evaluate the L1 signalling in the preamble in order to determine additional parameters (particularly notch positions) as well as the final tuning frequency of the (dependent) Static Data Slice. |
| 0b11 | reserved for future use. |

*active_OFDM_symbol_duration***:** This 3-bit field indicates the duration of the active OFDM symbol according to table 117.

**Table 117: Active OFDM symbol duration coding**

| *active_OFDM_symbol_duration* | Description |
|---|---|
| 0b000 | 448 µs (4k Fast Fourier Transform (FFT) mode for 8 MHz Cable Television (CATV) systems) |
| 0b001 | 597,33 µs (4k FFT mode for 6 MHz CATV systems) |
| 0b010 to 0b111 | reserved for future use |

*guard_interval***:** This 3-bit field indicates the guard interval according to table 118.

**Table 118: C2 guard interval coding**

| guard_interval | Description |
|---|---|
| 0b000 | 1/128 |
| 0b001 | 1/64 |
| 0b0010 to 0b111 | reserved for future use |

## 6.4.6.2      SH delivery system descriptor

This descriptor (see table 119) is used to transmit the physical parameters for each DVB Satellite to Handheld Framing and Modulation (DVB-SH) (ETSI EN 302 583 [10]) signal in the DVB network. This descriptor appears exactly once for each DVB transport stream in each NIT sub_table describing the network. This descriptor is made of static and a dynamic part based on a modulation loop.

**Table 119: SH delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| SH_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    diversity_mode | 4 | bslbf |
|    reserved_future_use | 4 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       modulation_type | 1 | bslbf |
|       interleaver_presence | 1 | bslbf |
|       interleaver_type | 1 | bslbf |
|       reserved_future_use | 5 | bslbf |
|       if (modulation_type == 0b0) { | | |
|          polarization | 2 | bslbf |
|          roll_off | 2 | bslbf |
|          modulation_mode | 2 | bslbf |
|          code_rate | 4 | bslbf |
|          symbol_rate | 5 | bslbf |
|          reserved_future_use | 1 | bslbf |
|       } else { | | |
|          bandwidth | 3 | bslbf |
|          priority | 1 | bslbf |
|          constellation_and_hierarchy | 3 | bslbf |
|          code_rate | 4 | bslbf |
|          guard_interval | 2 | bslbf |
|          transmission_mode | 2 | bslbf |
|          common_frequency | 1 | bslbf |
|       } | | |
|       if (interleaver_presence == 0b1) { | | |
|          if (interleaver_type == 0b0) { | | |
|             common_multiplier | 6 | uimsbf |
|             nof_late_taps | 6 | uimsbf |
|             nof_slices | 6 | uimsbf |
|             slice_distance | 8 | uimsbf |
|             non_late_increments | 6 | uimsbf |
|          } else { | | |
|             common_multiplier | 6 | uimsbf |
|             reserved_future_use | 2 | uimsbf |
|          } | | |
|       } | | |
|    } | | |
| } | | |

Semantics for the SH delivery system descriptor:

*diversity_mode***:** This 4-bit field describes the diversity modes and is coded according to table 120.

**Table 120: Diversity mode coding**

| diversity_mode | paTS | FEC diversity | FEC at PHY | FEC at link |
|---|---|---|---|---|
| 0b0000 | no | no | no | no |
| 0b0001 to 0b0111 | reserved for future use | | | |
| 0b1000 | yes | no | no | no |
| 0b1001 to 0b1100 | reserved for future use | | | |
| 0b1101 | yes | yes | no | yes |
| 0b1110 | yes | yes | yes | no |
| 0b1111 | yes | yes | yes | yes |

The modulation loop follows and has a number of iterations that depends on the selected SH configuration. For instance N=1 in Single Frequency Network (SFN), N=2 in non-SFN.

*modulation_type***:** This 1-bit field indicates which modulation is being used according to table 121.

**Table 121: SH modulation type coding**

| modulation_type | Description |
|---|---|
| 0b0 | Time-Domain Multiplex (TDM) |
| 0b1 | OFDM |

*interleaver_presence***:** This 1-bit field indicates the use of an interleaver according to table 122.

**Table 122: Interleaver presence coding**

| interleaver_presence | Description |
|---|---|
| 0b0 | no interleaver info follows |
| 0b1 | an interleaver info follows |

*interleaver_type***:** This 1-bit field indicates which interleaver is being used. Two options are possible:

- a complete interleaver describes the interleaver in the same manner as the Transmission Parameter Signalling (TPS) or signalling field;

- a short interleaver assumes that the interleaver is a uniform one and therefore gives the *common_multiplier* only.

*polarization***:** The *polarization* is a 2-bit field specifying the polarization of the transmitted signal. The first bit defines whether the polarization is linear or circular (see table 123).

**Table 123: Polarization coding**

| polarization | Description |
|---|---|
| 0b00 | linear - horizontal |
| 0b01 | linear - vertical |
| 0b10 | circular - left |
| 0b11 | circular - right |

*roll_off***:** This 2-bit field indicates the chosen roll-off factor according to table 124.

**Table 124: Roll-off factor**

| roll_off | Description |
|---|---|
| 0b00 | α = 0,35 |
| 0b01 | α = 0,25 |
| 0b10 | α = 0,15 |
| 0b11 | reserved for future use |

*modulation_mode***:** This 2-bit field indicates the chosen modulation according to table 125.

**Table 125: Modulation mode coding**

| *modulation_mode* | Description |
|---|---|
| 0b00 | QPSK |
| 0b01 | 8PSK |
| 0b10 | 16-ary Amplitude and Phase Shift Keying (16APSK) |
| 0b11 | reserved for future use |

*code_rate***:** This 4-bit field indicates the chosen code rate according to table 126.

**Table 126: Code rate coding**

| *code_rate* | Description |
|---|---|
| 0b0000 | 1/5 standard |
| 0b0001 | 2/9 standard |
| 0b0010 | 1/4 standard |
| 0b0011 | 2/7 standard |
| 0b0100 | 1/3 standard |
| 0b0101 | 1/3 complementary |
| 0b0110 | 2/5 standard |
| 0b0111 | 2/5 complementary |
| 0b1000 | 1/2 standard |
| 0b1001 | 1/2 complementary |
| 0b1010 | 2/3 standard |
| 0b1011 | 2/3 complementary |
| 0b1100 to 0b1111 | reserved for future use |

*symbol_rate***:** This 5-bit field signals the TDM symbol rate expressed in Msymbols per second according to table 127. In case OFDM modulation is also used, the *symbol_rate* shall also be compliant with the corresponding symbol rate table for single carrier TDM in ETSI TS 102 772 [23] where the OFDM parameters (bandwidth and guard interval) and the TDM roll-off determine a TDM symbol rate.

**Table 127: Symbol rate coding**

| *symbol_rate* | Equivalent bandwidth | Equivalent guard interval | Symbol rate for $\alpha = 0{,}15$ | Symbol rate for $\alpha = 0{,}25$ | Symbol rate for $\alpha = 0{,}35$ |
|---|---|---|---|---|---|
| 0b0 0000 | 8 | 1/4 | 34/5 | 32/5 | 29/5 |
| 0b0 0001 | 8 | 1/8 | 62/9 | 56/9 | 52/9 |
| 0b0 0010 | 8 | 1/16 | 116/17 | 108/17 | 100/17 |
| 0b0 0011 | 8 | 1/32 | 224/33 | 208/33 | 64/11 |
| 0b0 0100 | 7 | 1/4 | 119/20 | 28/5 | 203/40 |
| 0b0 0101 | 7 | 1/8 | 217/36 | 49/9 | 91/18 |
| 0b0 0110 | 7 | 1/16 | 203/34 | 189/34 | 175/34 |
| 0b0 0111 | 7 | 1/32 | 196/33 | 182/33 | 56/11 |
| 0b0 1000 | 6 | 1/4 | 51/10 | 24/5 | 87/20 |
| 0b0 1001 | 6 | 1/8 | 31/6 | 14/3 | 13/3 |
| 0b0 1010 | 6 | 1/16 | 87/17 | 81/17 | 75/17 |
| 0b0 1011 | 6 | 1/32 | 56/11 | 52/11 | 48/11 |
| 0b0 1100 | 5 | 1/4 | 17/4 | 4/1 | 29/8 |
| 0b0 1101 | 5 | 1/8 | 155/36 | 35/9 | 65/18 |
| 0b0 1110 | 5 | 1/16 | 145/34 | 135/34 | 125/34 |
| 0b0 1111 | 5 | 1/32 | 140/33 | 130/33 | 40/11 |
| 0b1 0000 | 1,7 | 1/4 | 34/25 | 32/25 | 29/25 |
| 0b1 0001 | 1,7 | 1/8 | 62/45 | 56/45 | 52/45 |
| 0b1 0010 | 1,7 | 1/16 | 116/85 | 108/85 | 20/17 |
| 0b1 0011 | 1,7 | 1/32 | 224/165 | 208/165 | 64/55 |
| 0b1 0100 to 0b1 1111 | reserved for future use | | | | |

*bandwidth***:** This 3-bit field gives the OFDM bandwidth according to table 128.

**Table 128: Bandwidth coding**

| bandwidth | Description |
|---|---|
| 0b000 | 8 MHz |
| 0b001 | 7 MHz |
| 0b010 | 6 MHz |
| 0b011 | 5 MHz |
| 0b100 | 1,7 MHz |
| 0b101 to 0b111 | reserved for future use |

**priority:** This 1-bit field indicates the stream's hierarchical priority according to table 129. Its semantics depend on the chosen *constellation_and_hierarchy* value.

**Table 129: Priority coding**

| constellation_and_hierarchy | priority | Description |
|---|---|---|
| 0b000 to 0b001 | 0b0 | n/a |
|  | 0b1 | no priority mode |
| 0b010 to 0b100 | 0b0 | LP |
|  | 0b1 | HP |

**constellation_and_hierarchy:** This 3-bit field indicates the stream constellation and hierarchy according to table 130.

**Table 130: Constellation and hierarchy coding**

| constellation_and_hierarchy | Description |
|---|---|
| 0b000 | QPSK |
| 0b001 | 16QAM, non-hierarchical |
| 0b010 | 16QAM, hierarchical, α = 1 |
| 0b011 | 16QAM, hierarchical, α = 2 |
| 0b100 | 16QAM, hierarchical, α = 3 |
| 0b101 to 0b111 | reserved for future use |

**guard_interval:** This 2-bit field gives the chosen guard interval according to table 131.

**Table 131: Guard interval coding**

| guard_interval | Description |
|---|---|
| 0b00 | 1/32 |
| 0b01 | 1/16 |
| 0b10 | 1/8 |
| 0b11 | 1/4 |

**transmission_mode:** This 2-bit field indicates the transmission mode according to table 132.

**Table 132: Transmission mode coding**

| transmission_mode | Description |
|---|---|
| 0b00 | 1k mode |
| 0b01 | 2k mode |
| 0b10 | 4k mode |
| 0b11 | 8k mode |

**common_frequency:** This 1-bit field indicates whether the modulation is used over a common frequency. A value of 0b0 denotes that this is not a common frequency. A value of 0b1 indicates that this is a common frequency.

**common_multiplier:** This 6-bit field indicates the length increment in Interleaving Units between two consecutive taps of the physical interleaver belonging to the late tap part. The *common_multiplier* is also used to compute the actual tap length increment in the non-late category.

*nof_late_taps*: This 6-bit field indicates the number of taps of the physical time interleaver that belong to the late tap part. Values vary from 0 to 48: 0 signals "no taps belong to the late tap part", 48 signals "48 taps belong to late tap part".

*nof_slices*: This 6-bit field indicates the number of slices over which the physical time interleaver spans. Minimum value is 1, in which case all taps belong to the late tap part.

*slice_distance*: This 8-bit field indicates in units of Satellite services to Handheld devices (SH) frames the distance between two consecutive slices of the physical time interleaver. Values vary from 1 to 63; if taps are all in the late tap part, this value shall be set to 1.

*non_late_increments*: This 6-bit field indicates the length increment between two consecutive taps belonging to the same non-late slice of the physical interleaver. The actual length increment in Interleaving Units is computed by multiplying this field with the value of the *common_multiplier* field.

## 6.4.6.3    T2 delivery system descriptor

The `T2_delivery_system_descriptor` (see table 133) shall be used in the TS loop of the NIT to describe DVB-T2 transmissions according to ETSI EN 302 755 [11]. This descriptor maps DVB transport streams to data PLPs and DVB-T2 systems. For each DVB transport stream carried in a DVB-T2 system in the network, there shall be at least one `T2_delivery_system_descriptor` in the corresponding TS loop of the respective NIT sub_table. The descriptor reflects a mapping of a TS - heading the NIT TS loop - to a data PLP, whereby the *T2_system_id* field might occur multiple times, via multiple instances of the descriptor, because a particular combination of DVB transport stream and data PLP may belong to different DVB-T2 systems. Within a network, several different DVB transport streams may map to the same combination of *T2_system_id* and *plp_id* of the data PLP, for example when the same DVB-T2 system is used throughout a network and in different geographical areas different DVB transport streams are used and mapped to the same data PLP.

If this descriptor is present in the NIT, the following definitions apply:

- The *transport_stream_id* and *original_network_id* announced within the TS loop of the NIT identify the DVB transport stream that is carried by the (PLP, T2 system) pair identified with *plp_id* and *T2_system_id* in the descriptor.

NOTE 1:  In the case of multiple PLPs in a T2 multiplex and in the presence of a common PLP, the corresponding TS is split into a data PLP, being referred to by the *plp_id*, and a common PLP. A corresponding re-assembly operation of the data PLP and common PLP on the receiver side is performed to recover the TS to be output by the receiver.

**Table 133: T2 delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| T2_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    plp_id | 8 | uimsbf |
|    T2_system_id | 16 | uimsbf |
|    if (descriptor_length > 4) { | | |
|       SISO_MISO | 2 | bslbf |
|       bandwidth | 4 | bslbf |
|       reserved_future_use | 2 | bslbf |
|       guard_interval | 3 | bslbf |
|       transmission_mode | 3 | bslbf |
|       other_frequency_flag | 1 | bslbf |
|       tfs_flag | 1 | bslbf |
|       for (i=0;i<N;i++) { | | |
|          cell_id | 16 | uimsbf |
|          if (tfs_flag == 0b1) { | | |
|             frequency_loop_length | 8 | uimsbf |
|             for (j=0;j<N;j++) { | | |
|                centre_frequency | 32 | uimsbf |
|             } | | |
|          } else { | | |
|             centre_frequency | 32 | uimsbf |
|          } | | |
|          subcell_info_loop_length | 8 | uimsbf |
|          for (j=0;j<N;j++) { | | |
|             cell_id_extension | 8 | uimsbf |
|             transposer_frequency | 32 | uimsbf |
|          } | | |
|       } | | |
|    } | | |
| } | | |

Semantics for the T2 delivery system descriptor:

**plp_id:** This 8-bit field identifies uniquely a PLP within the DVB-T2 system, identified with the *T2_system_id*. The term is defined in ETSI EN 302 755 [11].

**T2_system_id:** This 16-bit field identifies uniquely the DVB-T2 system within the DVB network (identified by *network_id*). The term is defined in ETSI EN 302 755 [11].

The remaining part of this descriptor, immediately following the *T2_system_id* field, may be present at most once per DVB-T2 system, because the parameters are uniquely applicable to all DVB transport streams carried over a particular DVB-T2 system. The presence or absence of that part can be derived from the *descriptor_length* field. In the absence of the remaining part this length equals 0x04, otherwise a larger value applies.

*SISO_MISO:* This 2-bit field indicates the SISO/MISO mode according to table 134.

**Table 134: SISO/MISO mode coding**

| *SISO_MISO* | Description |
|---|---|
| 0b00 | Single Input, Single Output (SISO) |
| 0b01 | Multiple Input, Single Output (MISO) |
| 0b10 to 0b11 | reserved for future use |

**bandwidth:** This 4-bit field indicates the bandwidth in use according to table 135.

**Table 135: Bandwidth coding**

| bandwidth | Description |
|---|---|
| 0b0000 | 8 MHz |
| 0b0001 | 7 MHz |
| 0b0010 | 6 MHz |
| 0b0011 | 5 MHz |
| 0b0100 | 10 MHz |
| 0b0101 | 1,712 MHz |
| 0b0110 to 0b1111 | reserved for future use |

*guard_interval***:** This 3-bit field indicates the guard interval according to table 136.

**Table 136: Guard interval coding**

| guard_interval | Description |
|---|---|
| 0b000 | 1/32 |
| 0b001 | 1/16 |
| 0b010 | 1/8 |
| 0b011 | 1/4 |
| 0b100 | 1/128 |
| 0b101 | 19/128 |
| 0b110 | 19/256 |
| 0b111 | reserved for future use |

*transmission_mode***:** This 3-bit field indicates the FFT size of the signals transmitted within the associated cell according to table 137.

**Table 137: Transmission mode coding**

| transmission_mode | Description |
|---|---|
| 0b000 | 2k mode |
| 0b001 | 8k mode |
| 0b010 | 4k mode |
| 0b011 | 1k mode |
| 0b100 | 16k mode |
| 0b101 | 32k mode |
| 0b110 to 0b111 | reserved for future use |

*other_frequency_flag***:** This 1-bit flag indicates whether other frequencies (non-Time Frequency Slicing (TFS) case) or other groups of frequencies (TFS case) are in use. The value 0b0 indicates that the set of frequencies (non-TFS case) or the set of groups of frequencies (TFS case) included in the descriptor is complete, whereas the value 0b1 indicates that the set is incomplete.

*tfs_flag***:** This 1-bit flag indicates whether a TFS arrangement is in place or not. It shall be coded according to table 138.

**Table 138: TFS flag coding**

| tfs_flag | Description |
|---|---|
| 0b0 | no TFS arrangement in place |
| 0b1 | TFS arrangement in place |

*cell_id***:** This 16-bit field uniquely identifies a cell, as defined in ETSI EN 302 755 [11].

 NOTE 2: *cell_id* = 0x0000 indicates that no *cell_id* is assigned. In this case information for different cells can still be provided by means of multiple instances of the cell loop.

*frequency_loop_length***:** This 8-bit field indicates the total length in bytes of the following loop that enumerates two to six centre frequencies belonging to the TFS arrangement associated with the named *cell_id* and its parameters. This loop is present only in the case of a TFS arrangement in place, otherwise only a single frequency per cell will be provided with a single instance of the *centre_frequency* parameter.

*centre_frequency*: This 32-bit field indicates the centre frequency value in multiples of 10 Hz. The coding range is from minimum 10 Hz (0x0000 0001) up to a maximum of 42 949 672 950 Hz (0xFFFF FFFF).

*subcell_info_loop_length*: This 8-bit field conveys the total length in bytes of the following loop that describes the subcells.

*cell_id_extension*: This 8-bit field is used to identify a subcell within a cell.

*transposer_frequency*: This 32-bit field indicates the centre frequency that is used by a transposer in the sub-cell indicated. It is encoded in the same way as the *centre_frequency* field.

## 6.4.6.4       C2 bundle delivery system descriptor

The `C2_bundle_delivery_system_descriptor` (see table 139) shall be used in the TS loop of the NIT to describe DVB-C2 transmissions according to ETSI EN 302 769 [9] using channel bundling. This descriptor maps DVB transport streams bundled within multiple data PLPs, data slices, and DVB-C2 systems. The `C2_bundle_delivery_system_descriptor` shall list the given parameters of all bundled PLPs which are required to reassemble the TS.

NOTE 1:  Bundled PLPs may be transmitted via several DVB-C2 systems.

NOTE 2:  Different terminology is used in ETSI EN 302 769 [9] for functionally equivalent capabilities.

Each TS loop of the NIT describing a TS which is conveyed within a DVB-C2 system, shall contain either a `C2_delivery_system_descriptor` (see clause 6.4.6.1), or one or more `C2_bundle_delivery_system_descriptor`. When multiple `C2_bundle_delivery_system_descriptor` are present in the same TS loop of the NIT describing a TS, then the bundle is defined by the combination of the PLPs listed in all those descriptors.

**Table 139: C2 bundle delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `C2_bundle_delivery_system_descriptor() {` | | |
| `    descriptor_tag` | 8 | uimsbf |
| `    descriptor_length` | 8 | uimsbf |
| `    descriptor_tag_extension` | 8 | uimsbf |
| `    for (i=0;i<N;i++) {` | | |
| `        plp_id` | 8 | uimsbf |
| `        data_slice_id` | 8 | uimsbf |
| `        C2_System_tuning_frequency` | 32 | bslbf |
| `        C2_System_tuning_frequency_type` | 2 | uimsbf |
| `        active_OFDM_symbol_duration` | 3 | bslbf |
| `        guard_interval` | 3 | bslbf |
| `        primary_channel` | 1 | bslbf |
| `        reserved_zero_future_use` | 7 | bslbf |
| `    }` | | |
| `}` | | |

Semantics for the C2 bundle delivery system descriptor:

*plp_id*: This 8-bit field shall be coded according to clause 6.4.6.1.

*data_slice_id*: This 8-bit field shall be coded according to clause 6.4.6.1.

*C2_System_tuning_frequency*: This 32-bit field shall be coded according to clause 6.4.6.1.

*C2_System_tuning_frequency_type*: This 2-bit field shall be coded according to clause 6.4.6.1.

*active_OFDM_symbol_duration*: This 3-bit field shall be coded according to clause 6.4.6.1.

*guard_interval*: This 3-bit field shall be coded according to clause 6.4.6.1.

*primary_channel*: This 1-bit field indicates whether the associated PLP is using the primary clock for the generation of the Input Stream SYnchronizer (ISSY) time stamps. If the associated PLP uses the primary clock for ISSY generation, this field shall be set to 0b1. Otherwise this field shall be set to 0b0.

NOTE 3: Older versions of the present document used different terminology for primary channel and primary clock.

## 6.4.6.5    S2X satellite delivery system descriptors

### 6.4.6.5.1    Introduction

When DVB Satellite Framing and Modulation, Second Generation Extensions (DVB-S2X) modulation as defined in ETSI EN 302 307-2 [8] is used, there are 2 possibilities to describe the delivery system, using the `S2X_satellite_delivery_system_descriptor` or the `S2Xv2_satellite_delivery_system_descriptor`. For `S2X_satellite_delivery_system_descriptor` see clause 6.4.6.5.2. For `S2Xv2_satellite_delivery_system_descriptor` see clause 6.4.6.5.3.

When the transponder is used in a beamhopping mode then the `S2Xv2_satellite_delivery_system_descriptor` shall be used.

Transponders which require the `S2Xv2_satellite_delivery_system_descriptor` (e.g. non-geostationary satellites) will exclude older receivers. Existing receivers where support of these transponders is wanted, will have to add support for the `S2Xv2_satellite_delivery_system_descriptor`, otherwise the transponders will be ignored.

For transponders compatible with the `S2X_satellite_delivery_system_descriptor`, using that descriptor will allow to include the receivers that do not support `S2Xv2_satellite_delivery_system_descriptor`.

In order to support receivers that do not support the `S2Xv2_satellite_delivery_system_descriptor`, the `S2X_satellite_delivery_system_descriptor` shall be used.

### 6.4.6.5.2    S2X satellite delivery system descriptor (version 1)

In case the delivery system is described by `S2X_satellite_delivery_system_descriptor`, `S2X_satellite_delivery_system_descriptor` is the only delivery system descriptor needed. It shall be coded according to table 140. It describes the modulation parameters used for DVB-S2X transmissions. When channel bonding is used, it also describes the modulation parameters of the channels that are bonded with the primary channel. The PSI/SI tables as are required by the present document, may need to be carried in some or all of the bonded channels. To support network discovery by receivers, all bonded channels shall convey at least a NIT for the actual network. The distribution and replication of the further PSI/SI tables for a TS across all the bonded channels is subject to operational decisions by the network operator.

**Table 140: S2X satellite delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| S2X_satellite_delivery_system_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    receiver_profiles | 5 | bslbf |
|    reserved_zero_future_use | 3 | bslbf |
|    S2X_mode | 2 | uimsbf |
|    scrambling_sequence_selector | 1 | bslbf |
|    reserved_zero_future_use | 3 | bslbf |
|    TS_GS_S2X_mode | 2 | bslbf |
|    if (scrambling_sequence_selector == 0b1) { | | |
|       reserved_zero_future_use | 6 | bslbf |
|       scrambling_sequence_index | 18 | uimsbf |
|    } | | |
|    frequency (see note) | 32 | bslbf |
|    orbital_position (see note) | 16 | bslbf |
|    west_east_flag (see note) | 1 | bslbf |
|    polarization (see note) | 2 | bslbf |
|    multiple_input_stream_flag (see note) | 1 | bslbf |
|    reserved_zero_future_use | 1 | bslbf |
|    roll_off (see note) | 3 | bslbf |
|    reserved_zero_future_use | 4 | bslbf |
|    symbol_rate (see note) | 28 | bslbf |
|    if (multiple_input_stream_flag == 0b1) { | | |
|       input_stream_identifier (see note) | 8 | uimsbf |
|    } | | |
|    if (S2X_mode == 2) { | | |
|       timeslice_number | 8 | uimsbf |
|    } | | |
|    if (S2X_mode == 3) { | | |
|       reserved_zero_future_use | 7 | bslbf |
|       num_channel_bonds_minus_one | 1 | uimsbf |
|       for (i=0;i<N;i++) { | | |
|          frequency | 32 | bslbf |
|          orbital_position | 16 | bslbf |
|          west_east_flag | 1 | bslbf |
|          polarization | 2 | bslbf |
|          bonded_channel_multiple_input_stream_flag | 1 | bslbf |
|          reserved_zero_future_use | 1 | bslbf |
|          roll_off | 3 | bslbf |
|          reserved_zero_future_use | 4 | bslbf |
|          symbol_rate | 28 | bslbf |
|          if (bonded_channel_multiple_input_stream_flag == 0b1) { | | |
|             input_stream_identifier | 8 | uimsbf |
|          } | | |
|       } | | |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       reserved_future_use | 8 | bslbf |
|    } | | |
| } | | |
| NOTE:    When channel bonding is used, these parameters describe the primary channel. | | |

Semantics for the S2X satellite delivery system descriptor:

*receiver_profiles***:** This 5-bit field indicates which receiver profiles are targeted by the stream. It shall be coded according to table 141. When no particular receiver profile is targeted, all bits shall be set to zero.

**Table 141: Receiver profiles coding**

| receiver_profiles bit | Description |
|---|---|
| $b_0$ (see note) | broadcast services |
| $b_1$ | interactive services |
| $b_2$ | DSNG services |
| $b_3$ | professional services |
| $b_4$ | Very Low Signal to Noise Ratio (VL-SNR) services |
| NOTE: This bit is transmitted last (see clause 5.1.6). | |

**S2X_mode:** This 2-bit field indicates in which DVB-S2X mode the stream is operated. It shall be coded according to table 142.

**Table 142: S2X mode coding**

| S2X_mode | Description |
|---|---|
| 0 | reserved for future use |
| 1 | S2X |
| 2 | S2X + time slicing |
| 3 | S2X + channel bonding |

**scrambling_sequence_selector:** This 1-bit field shall be coded according to clause 6.2.13.3.

**TS_GS_S2X_mode:** This 2-bit field indicates in which mode the stream is operated. It shall be coded according to table 143.

**Table 143: TS/GS S2X mode coding**

| TS_GS_S2X_mode (see note) | Description |
|---|---|
| 0 | generic packetized |
| 1 | GSE |
| 2 | GSE high efficiency mode |
| 3 | DVB transport stream |
| NOTE: These values are compatible with the coding of the TS/GS field in the BBFrame header of DVB-S2X (see clause 5.1.6 of ETSI EN 302 307-2 [8]). | |

**scrambling_sequence_index:** This 18-bit field shall be coded according to clause 6.2.13.3.

**frequency:** This 32-bit field shall be coded according to clause 6.2.13.2.

**orbital_position:** This 16-bit field shall be coded according to clause 6.2.13.2.

**west_east_flag:** This 1-bit field shall be coded according to clause 6.2.13.2.

**polarization:** This 2-bit field shall be coded according to clause 6.2.13.2.

**multiple_input_stream_flag:** This 1-bit field shall be coded according to clause 6.2.13.3.

**roll_off:** This 3-bit field indicates the roll-off factor used in DVB-S2X. It shall be coded according to table 144.

**Table 144: S2X roll off coding**

| roll_off | Description |
|---|---|
| 0b000 | α = 0,35 |
| 0b001 | α = 0,25 |
| 0b010 | α = 0,20 |
| 0b011 | reserved for future use |
| 0b100 | α = 0,15 |
| 0b101 | α = 0,10 |
| 0b110 | α = 0,05 |
| 0b111 | reserved for future use |

*symbol_rate*: This 28-bit field shall be coded according to clause 6.2.13.2.

*input_stream_identifier*: This 8-bit field shall be coded according to clause 6.2.13.3.

*timeslice_number*: This 8-bit field shall be coded according to clause 6.2.13.3.

*num_channel_bonds_minus_one*: This 1-bit field indicates the number of descriptions of channels bonded with the master channel that follow. If one channel is bonded with the primary channel, this field shall be set to zero, and one instance of the description loop shall follow. If two channels are bonded with the primary channel, this field shall be set to one, and two instances of the description loop shall follow.

 NOTE: The total number of bonded channels, including the primary channel, is hence the value of this field plus two.

*bonded_channel_multiple_input_stream_flag*: This 1-bit field shall be coded as the *multiple_input_stream_flag* field defined in clause 6.2.13.3 to describe each non-primary bonded channel.

## 6.4.6.5.3          S2Xv2 satellite delivery system descriptor (version 2)

In most cases (where channel bond is set to 0) the delivery system is described by a single
`S2Xv2_satellite_delivery_system_descriptor`.

When channel bonding is used, one `S2Xv2_satellite_delivery_system_descriptor` describes the primary (channel bond is set to 1) and one or two `S2Xv2_satellite_delivery_system_descriptor` describes the secondary (channel bond is set to 2).

The `S2Xv2_satellite_delivery_system_descriptor` shall be coded according to table 144a. It describes the modulation parameters used for DVB-S2X transmissions. The PSI/SI tables as are required by the present document, may need to be carried in some or all of the bonded channels. To support network discovery by receivers, all bonded channels shall convey at least a NIT for the actual network. The distribution and replication of the further PSI/SI tables for a TS across all the bonded channels is subject to operational decisions by the network operator.

**Table 144a: S2Xv2 satellite delivery system descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `S2Xv2_satellite_delivery_system_descriptor() {` | | |
| `   descriptor_tag` | 8 | uimsbf |
| `   descriptor_length` | 8 | uimsbf |
| `   descriptor_tag_extension` | 8 | uimsbf |
| `   S2Xv2_satellite_delivery_system_info()` | | |
| `}` | | |

Semantics for the S2X v2 satellite delivery system descriptor:

`S2Xv2_satellite_delivery_system_info`: The content of this field shall be coded according to table 144b.

**Table 144b: S2Xv2 satellite delivery system info**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `S2Xv2_satellite_delivery_system_info() {` | | |
| `    delivery_system_id` | 32 | uimsbf |
| `    S2Xv2_mode` | 4 | uimsbf |
| `    multiple_input_stream_flag` | 1 | bslbf |
| `    roll_off` | 3 | bslbf |
| `    reserved_zero_future_use` | 2 | bslbf |
| `    NCR_reference` | 1 | bslbf |
| `    NCR_version` | 1 | bslbf |
| `    channel_bond` | 2 | uimsbf |
| `    polarization` | 2 | bsblf |
| `    if (S2Xv2_mode == 1 or S2Xv2_mode == 2) {` | | |
| `        scrambling_sequence_selector` | 1 | bslbf |
| `    } else {` | | |
| `        reserved_zero_future_use` | 1 | bslbf |
| `    }` | | |
| `    TS_GS_S2X_mode` | 2 | bslbf |
| `    receiver_profiles` | 5 | bslbf |
| `    satellite_id` | 24 | uimsbf |
| `    frequency` | 32 | bslbf |
| `    symbol_rate` | 32 | bslbf |
| `    if (multiple_input_stream_flag == 1) {` | | |
| `        input_stream_identifier` | 8 | uimsbf |
| `    }` | | |
| `    if (S2Xv2_mode == 1 or S2Xv2_mode == 2) {` | | |
| `        if (scrambling_sequence_selector == 1) {` | | |
| `            reserved_zero_future_use` | 6 | bslbf |
| `            scrambling_sequence_index` | 18 | uimsbf |
| `        }` | | |
| `    }` | | |
| `    if (S2Xv2_mode == 2 or S2Xv2_mode == 5) {` | | |
| `        timeslice_number` | 8 | uimsbf |
| `    }` | | |
| `    if (channel_bond == 1) {` | | |
| `        reserved_zero_future_use` | 7 | bslbf |
| `        num_channel_bonds_minus_one` | 1 | uimsbf |
| `        for (i=0;i<N;i++) {` | | |
| `            secondary_delivery_system_id` | 32 | uimsbf |
| `        }` | | |
| `    }` | | |
| `    if (S2Xv2_mode == 4 or S2Xv2_mode == 5) {` | | |
| `        SOSF_WH_sequence_number` | 8 | uimsbf |
| `        SFFI_selector` | 1 | bsblf |
| `        beam_hopping_time_plan_selector` | 1 | bsblf |
| `        reserved_zero_future_use` | 2 | bsblf |
| `        reference_scrambing_index` | 20 | uimsbf |
| `        if (SFFI_selector == 1) {` | | |
| `            SFFI` | 4 | bslf |
| `        } else {` | | |
| `            reserved_zero_future_use` | 4 | bsblf |
| `        }` | | |
| `        payload_scrambling_index` | 20 | uimsbf |
| `        if (beam_hopping_time_plan_selector == 1) {` | | |
| `            beamhopping_time_plan_id` | 32 | uimsbf |
| `        }` | | |
| `        superframe_pilots_WH_sequence_number` | 5 | uimsbf |
| `        postamble_PLI` | 3 | bslf |
| `    }` | | |
| `    for (i=0;i<N;i++) {` | | |
| `        reserved_zero_future_use` | 8 | bslbf |
| `    }` | | |
| `}` | | |

Semantics for the S2X v2 satellite delivery system info:

*delivery_system_id***:** This 32-bit field unique identifier identifies this delivery system.

NOTE 1: In case of channel bonding, the channel bond is identified by the primary delivery system descriptor with this *delivery_system_id*.

*S2Xv2_mode***:** This 4-bit field indicates in which DVB-S2X mode the stream is operated. It shall be coded according to table 144c.

**Table 144c: S2Xv2 mode coding**

| S2Xv2_mode | Description |
|---|---|
| 0 | reserved for future use |
| 1 | S2X |
| 2 | S2X + time slicing |
| 3 | reserved for future use |
| 4 | S2X superframe (Annex E of ETSI EN 302 307-2 [8]) |
| 5 | S2X superframe (Annex E of ETSI EN 302 307-2 [8]) + timeslicing (Annex M of ETSI EN 302 307-1 [7]) |
| 6-15 | reserved for future use |

NOTE 2: In case of S2X superframe, the superframe format 4,5,6,7 have timeslicing mandatory so S2Xv2_mode will be 5. The superframe format 2,3 do not have timeslicing so S2Xv2_mode will be 4. The superframe format 0 and 1 have timeslicing optional, so S2Xv2_mode could be either 4 or 5. In case there is a conflict between superframe format received in the SFFI and the signalled S2Xv2_mode, then SFFI prevails.

*multiple_input_stream_flag***:** This 1-bit field shall be coded according to clause 6.2.13.3.

*roll_off***:** This 3-bit field shall be coded according to clause 6.4.6.5.2.

*NCR_reference*: This 1 bit field indicates which is the forward link timing reference symbol indicated by the NCR value, as specified in ETSI EN 301 545-2 [56] It shall be coded according to table 144c1.

**Table 144c1: NCR reference**

| NCR_reference | Description |
|---|---|
| 0 | The first symbol of the Start Of Frame field of the relevant DVB-S2 or DVB-S2X physical layer frame |
| 1 | The first symbol of the Start Of Superframe (SOSF) field of the ETSI EN 302 307-2 [8], annex E superframe |

*NCR_version***:** This 1-bit field indicates in which NCR version, as specified in ETSI EN 301 545-2 [56] is used on this delivery system. It shall be coded according to table 144d.

**Table 144d: NCR version coding**

| NCR_version | Description |
|---|---|
| 0 | NCR |
| 1 | NCR_v2 |

*channel_bond***:** This 2-bit field indicates if this delivery system is part of a channel bond and if yes, what role it has. It shall be coded according to table 144e.

**Table 144e: channel bond coding**

| channel_bond | Description |
|---|---|
| 0 | Not part of a channel bond |
| 1 | Channel bond primary |
| 2 | Channel bond secondary |
| 3 | Reserved for future use |

*polarization*: This 2-bit field shall be coded according to clause 6.2.13.3.

*scrambling_sequence_selector*: This 1-bit field shall be coded according to clause 6.2.13.3.

*TS_GS_S2X_mode*: This 2-bit field shall be coded according to clause 6.4.6.5.2.

*receiver_profiles*: This 5-bit field shall be coded according to clause 6.4.6.5.2.

*satellite_id*: This 24-bit field contains the *satellite_id* of the satellite from which the signal is transmitted to the receiver. The Satellite Access Table with a *satellite_table_id* of 0 contains the details of the satellite position for the *satellite_id*.

*frequency*: This 32-bit field shall be coded according to clause 6.2.13.3.

*symbol_rate*: This 32-bit field indicates the symbol_rate giving the 4-bit BCD values specifying 8 characters of the symbol_rate in Msymbols/s where the decimal point occurs after the fourth character (e.g. 0027,4500).

> EXAMPLE:   A symbol_rate value of 0x00274500 indicates a symbol rate of 27,4500 Msymbol/s.

*input_stream_identifier*: This 8-bit field shall be coded according to clause 6.2.13.3.

*scrambling_sequence_index*: This 18-bit field shall be coded according to clause 6.2.13.3.

*timeslice_number*: This 8-bit field shall be coded according to clause 6.2.13.3.

> NOTE 3:   For superframes of a superframe format that does not support timeslicing, this *timeslice_number* is not applicable.

*num_channel_bonds_minus_one*: This 1-bit field indicates the number of *delivery_system_id* of channels bonded with the primary channel that follow. If one channel is bonded with the primary channel, this field shall be set to 0, and one id shall follow. If two channels are bonded with the primary channel, this field shall be set to 1, and two ids of the description loop shall follow.

> NOTE 4:   The total number of bonded channels, including the primary channel, is hence the value of this field plus two.

*secondary_delivery_system_id*: This 32-bit field contains the *delivery_system_id* of the delivery system of a secondary channel that is bonded with this primary channel.

*SOSF_WH_sequence_number*: This 8-bit field shall contain the value of i to select the Walsh-Hadamard sequence for the superframe Start Of SuperFrame (SOSF) according to ETSI EN 302 307-2 [8], clause E.2.2.

*SFFI_selector*: This 1-bit field shall be set to 1 when there is only superframe format in use in the delivery system and it is provided in the SFFI field. Otherwise the field shall be set to 0.

*beam_hopping_time_plan_selector*: This 1-bit field shall be set to 1 when the *beamhopping_time_plan_id* is provided. Otherwise the field shall be set to 0.

> NOTE 5:   The beam hopping time plan for the *beam_hopping_time_plan_id* is provided in the Beamhopping Time Plan subtable of the SAT (see clause 5.2.11).

*reference_scrambling_index*: This 20-bit field shall be coded according to ETSI EN 302 307-2 [8], clause E.2.4.1.

*SFFI*: This 4-bit field shall contain the $b_{SFFI}$ that contains the superframe format according to ETSI EN 302 307-2 [8], clause E.2.3.

*payload_scrambling_index*: This 20-bit field shall be coded according to ETSI EN 302 307-2 [8], clause E.2.4.1.

*beamhopping_time_plan_id*: This 32-bit field contains the *beamhopping_time_plan_id* of the beamhopping time plan that is in use for this satellite delivery system.

*superframe_pilots_WH_sequence_number*: This 5-bit field contains the row number (counting from 0) of $H_{PilotA}$ that contains the Walsh-Hadamard sequence used for the pilot symbols as specified in ETSI EN 302 307-2 [8], clause E.3.1.1 and E.3.4.3 for the superframes of format 0,1,2,4,5,6 or 7. This 5-bit field contains the row number (counting from 0) of $H_{Pilot3}$ that contains the Walsh-Hadamard sequence used for the pilot symbols as specified in ETSI EN 302 307-2 [8], clause E.3.5.3 for the superframes of format 3.

*postamble_PLI*: This 2-bit field indicates the protection level required for the postamble, defined in clause E.3.7.8 of ETSI EN 302 307-2 [8], according to table 144f.

**Table 144f: Postamble PLI**

| PLI | Description |
|---|---|
| 0 | Identical to the Superframe PLI |
| 1 to 3 | Reserved |
| 4 | L= 180 symbols |
| 5 | L=360 symbols |
| 6 | L= 900 symbols |
| 7 | L = 90 symbols |

Where L is the postamble length as defined in clause E.3.7.8 of ETSI EN 302 307-2 [8].

## 6.4.7    Image icon descriptor

The image icon descriptor (see table 145) carries inline icon data or a Uniform Resource Locator (URL) that identifies the location of an icon file. It is intended to be used to carry or reference icons for two main purposes depending on where the icon is located:

1)    When found in the Related Content Table (RCT) (see ETSI TS 102 323 [21], clause 10) it shall be displayed as a call-to-action.

2)    When found in the NIT, BAT, SDT, EIT or SIT it may be used within a receiver's native user interface and position information shall be omitted.

The resolution and size of the image may be specified within the image encoding itself and/or by the icon type field. This will be determined depending on local profile.

**Table 145: Image icon descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| image_icon_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    descriptor_number | 4 | uimsbf |
|    last_descriptor_number | 4 | uimsbf |
|    reserved_future_use | 5 | uimsbf |
|    icon_id | 3 | uimsbf |
|    if (descriptor_number == 0x00) { | | |
|      icon_transport_mode | 2 | uimsbf |
|      position_flag | 1 | bslbf |
|      if (position_flag == 0b1 | | |
|        coordinate_system | 3 | uimsbf |
|        reserved_future_use | 2 | bslbf |
|        icon_horizontal_origin | 12 | uimsbf |
|        icon_vertical_origin | 12 | uimsbf |
|      } else { | | |
|        reserved_future_use | 5 | bslbf |
|      } | | |
|      icon_type_length | 8 | uimsbf |
|      for (i=0;i<N;i++) { | | |
|        icon_type_char | 8 | uimsbf |
|      } | | |
|      if (icon_transport_mode == 0) { | | |
|        icon_data_length | 8 | uimsbf |
|        for (i=0;i<N;i++) { | | |
|          icon_data_byte | 8 | uimsbf |
|        } | | |
|      } else if (icon_transport_mode == 1) { | | |
|        url_length | 8 | uimsbf |
|        for (i=0;i<N;i++) { | | |
|          url_char | 8 | uimsbf |
|        } | | |
|      } | | |
|    } else { | | |
|      icon_data_length | 8 | uimsbf |
|      for (i=0;i<N;i++) { | | |
|        icon_data_byte | 8 | uimsbf |
|      } | | |
|    } | | |
| } | | |

Semantics for the image icon descriptor:

*descriptor_number*: This 4-bit field gives the number of the descriptor. It is used to associate information which cannot be fitted into a single descriptor. The *descriptor_number* of the first of an associated set of descriptors shall be 0x00. The *descriptor_number* shall be incremented by 1 with each additional image_icon_descriptor with the same *icon_id* in this sub_table.

*last_descriptor_number*: This 4-bit field specifies the number of the last image_icon_descriptor (that is, the descriptor with the highest value of *descriptor_number*) of the associated set of descriptors of which this descriptor is part.

*icon_id*: This is a 3-bit field which identifies this icon as distinct from other icons delivered in the same descriptor loop. The scope of the id is restricted to the current descriptor loop. The *icon_id* value of 0b000 shall not be used in this descriptor (see ETSI TS 102 323 [21]).

*icon_transport_mode*: This 2-bit field indicates the mode of delivery of the icon as defined in table 146.

**Table 146: Icon transport mode coding**

| icon_transport_mode | Description |
|---|---|
| 0 | the icon is delivered in the *icon_data_byte* sequence of bytes |
| 1 | the location of the icon file is identified by the URL conveyed in the *url_char* sequence of bytes |
| 2 to 3 | reserved for future use |

*position_flag***:** This 1-bit field indicates whether on-screen position information is provided for the icon image. If this field is set to 0b1 then *icon_horizontal_origin* and *icon_vertical_origin* shall be used to position the icon on-screen. If this field is set to 0b0 the position is not specified.

NOTE 1:  If video scaling has occurred prior to the compositing of the video and graphics plane then the intended relationship between the icon position and underlying video may not be maintained.

*coordinate_system***:** This 3-bit field specifies the coordinate system on which the icon position is based. It shall be coded according to table 147.

**Table 147: Coordinate system coding**

| coordinate_system | Description |
|---|---|
| 0 | the coordinate system is 720 × 576 |
| 1 | the coordinate system is 1 280 × 720 |
| 2 | the coordinate system is 1 920 × 1 080 |
| 3 to 6 | reserved for future use |
| 7 | user defined |

NOTE 2:  If the coordinate system specified does not match the video resolution or display resolution then scaling of the icon position will be required. The mechanism for this is outside the scope of the present document.

*icon_horizontal_origin***:** The horizontal pixel position on the screen to be used as icon origin (top-left). When this field is set to zero the left most column of pixels of the icon shall be positioned in the left most pixel column of the display.

*icon_vertical_origin***:** The vertical pixel position on the screen to be used as icon origin (top-left). When this field is set to zero the upper most row of pixels of the icon shall be positioned in the upper most pixel row of the display.

*icon_type_length***:** The length of the icon type string. If the type is not specified then this value shall be 0.

*icon_type_length***:** This is an 8-bit field, a sequence of which conveys a string describing the image type. This shall be a Multipurpose Internet Mail Extensions (MIME) image media subtype as described in IETF RFC 2045 [32]. The top level media type "image" may be omitted. If the image types Portable Network Graphics (PNG), or Joint Photographic Experts Group (JPEG) are used then they shall conform to the restrictions defined in ETSI ES 201 812 [13], clause 15.

NOTE 3:  It is expected that receivers will implement at least the PNG and JPEG image formats conforming to the restrictions defined in IETF RFC 2045 [32], clause 15. Broadcasters are thus encouraged to restrict themselves to these formats.

EXAMPLE:     For a PNG image the sequence of *icon_type_char* fields would contain either `image/png` or, for short, `/png`.

*url_length***:** The length in bytes of the URL.

*url_char***:** This is an 8-bit field, a sequence of which conveys a URL which describes the location of an icon file. This field shall be encoded according to clause 6.2 of ETSI TS 102 323 [21]. The URL may be a DVB locator referencing a file in an object carousel as specified in clause 6.4 of ETSI TS 102 323 [21], or a reference to an IP-based resource.

*icon_data_length***:** The length in bytes of the icon data carried in the remainder of the descriptor.

## 6.4.8    Message descriptor

This descriptor (see table 148) allows broadcasters to provide receivers with a textual message which the receiver may display to the user at appropriate times (not defined here).

**Table 148: Message descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| message_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    message_id | 8 | uimsbf |
|    ISO_639_language_code | 24 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       text_char | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the message descriptor:

***message_id*:** This 8-bit field uniquely identifies the message within the scope of the sub_table in which it appears. Note that multiple descriptors with the same *message_id* shall not be broadcast unless they have differing *ISO_639_language_code* values. The value of this field shall be in the range of 1 to 255.

***ISO_639_language_code*:** This 24-bit field identifies the language of the textual message. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

***text_char*:** This is an 8-bit field, a sequence of which conveys the text to be displayed. Text information is coded using the character sets and methods described in annex A.

## 6.4.9 Network change notify descriptor

This descriptor (see table 149) allows broadcasters to signal network change events to receivers. A network change event is a single, clearly identifiable change in the network configuration, e.g. transmission parameters and/or available services, which may require action on the part of receivers.

Network changes may be confined to a part of the network, as indicated by the use of the *cell_id* field (used in DVB-T and DVB-T2 systems) or may apply to the whole network, as defined by the NIT.

Where used, the `network_change_notify_descriptor` shall signal at all times the current list of scheduled network change events. The absence of a `network_change_notify_descriptor` shall be used to indicate that there are no scheduled network change events.

**Table 149: Network change notify descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| network_change_notify_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       cell_id | 16 | uimsbf |
|       loop_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          network_change_id | 8 | uimsbf |
|          network_change_version | 8 | uimsbf |
|          start_time_of_change | 40 | bslbf |
|          change_duration | 24 | uimsbf |
|          receiver_category | 3 | uimsbf |
|          invariant_ts_present | 1 | bslbf |
|          change_type | 4 | uimsbf |
|          message_id | 8 | uimsbf |
|          if (invariant_ts_present == 0b1) { | | |
|             invariant_ts_tsid | 16 | uimsbf |
|             invariant_ts_onid | 16 | uimsbf |
|          } | | |
|       } | | |
|    } | | |
| } | | |

Semantics for the network change notify descriptor:

*cell_id*: This 16-bit field uniquely identifies a cell within a DVB-T or DVB-T2 network (as defined by *network_id*). A *cell_id* of 0x0000 shall be used to signal a change affecting all *cell_id*. Where multiple changes exist for a given *cell_id*, they shall be signalled in one instance of the descriptor. For other delivery systems, this field is not used and the network change applies to the network defined by the NIT carrying this descriptor.

*loop_length*: This 8-bit field specifies the length in bytes of the following items.

*network_change_id*: This 8-bit field is a unique identifier for the network change event signalled within this cell. No inferences should be made from the incremental or other nature of the value of *network_change_id*.

*network_change_version*: This 8-bit field signals the version of the change. It shall be changed if any parameter within this loop is changed. Values shall be assigned incrementally until 0xFF. If further *network_change_version* are required, a new *network_change_id* shall be issued.

*start_time_of_change*: This 40-bit field contains the time at which the network changes are planned to start in UTC and MJD (see annex C). This field is coded as 16 bits giving the 16 lsb of MJD followed by 24 bits coded as 6 digits in 4-bit BCD.

    EXAMPLE 1:    93/10/13 12:45:00 is coded as 0xC0 7912 4500.

*change_duration*: This 24-bit field indicates the duration of the network change in hours, minutes, and seconds coded as 6 digits in 4-bit BCD. Durations signalled within this field shall be limited to a maximum of 11:59:59.

    EXAMPLE 2:    01:45:30 is coded as 0x01 4530.

*receiver_category*: This 3-bit field indicates the category of receivers affected by the change being signalled according to table 150.

**Table 150: Receiver category coding**

| *receiver_category* | Description |
|---|---|
| 0 | all receivers |
| 1 | DVB-T2, or DVB-S2, or DVB-C2 capable receivers only |
| 2 to 7 | reserved for future use |

*invariant_ts_present*: If this 1-bit field is set to 0b1, an invariant DVB transport stream is being signalled. If set to 0b0, all multiplexes with this *cell_id* (for DVB-T or DVB-T2 systems) or within the network (for other delivery systems) should be considered as subject to change. An invariant DVB transport stream is one which uses the same transmission parameters before and after the network changes (but not necessarily during the change).

*change_type*: This 4-bit field specifies the type of change that will take place, as defined in table 151. Minor changes are defined as those changes which can be detected by a receiver by comparison of the old and new SI. Major changes are defined as those which could require a receiver to tune or scan away from the current multiplex.

The "default" category shall be used when another category does not adequately describe the current scenario, or when multiple categories would describe the current scenario. The "message only" category shall be used when there are no changes to the network but the broadcaster wishes to provide a message to be displayed by the receivers. The "coverage change" category shall be used when power and/or modulation parameter changes potentially change the coverage of a transmitter. It shall also be used when a cell or transmitter is being added or removed since this potentially also changes the coverage. A "coverage change" may result in the receiver no longer receiving signals from the most appropriate cell or transmitter.

**Table 151: Change type coding**

| change_type | Description |
|---|---|
| 0 | message only |
| 1 | minor - default |
| 2 | minor - multiplex removed |
| 3 | minor - service changed |
| 4 to 7 | reserved for future use for other minor changes |
| 8 | major - default |
| 9 | major - multiplex frequency changed |
| 10 | major - multiplex coverage changed |
| 11 | major - multiplex added |
| 12 to 15 | reserved for future use for other major changes |

*message_id*: This 8-bit field is used to link to a message in the `message_descriptor` carried in the same NIT. A *message_id* of 0x00 shall be used to signal that there is no message associated with this network change. Multiple network changes may be linked to a single *message_id*.

*invariant_ts_tsid*: This 16-bit field contains the *transport_stream_id* of the invariant DVB transport stream.

*invariant_ts_onid*: This 16-bit field contains the *original_network_id* of the invariant DVB transport stream.

## 6.4.10    Service relocated descriptor

If a service has moved from one TS to another, a mechanism is provided which enables an IRD to track the service at its new location (for example between DVB transport streams) by means of a `service_relocated_descriptor` (see table 152).

NOTE:    A new location for a service first of all implies new identifiers are being used. It does not necessarily imply that it has moved to a different physical location.

**Table 152: Service relocated descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_relocated_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    old_original_network_id | 16 | uimsbf |
|    old_transport_stream_id | 16 | uimsbf |
|    old_service_id | 16 | uimsbf |
| } | | |

Semantics for the service relocated descriptor:

***old_original_network_id*:** This 16-bit field contains the *original_network_id* of the TS in which the service was found before the relocation.

***old_transport_stream_id*:** This 16-bit field contains the *transport_stream_id* of the TS in which the service was found before the relocation.

***old_service_id*:** This 16-bit field contains the *service_id* of the service before the relocation.

## 6.4.11 Supplementary audio descriptor

The `supplementary_audio_descriptor` (see table 153) provides additional information about the associated audio component. Information from this and other descriptors allows the receiver to present the appropriate stream, or mix of streams, to the user. It shall be carried in the Elementary Stream (ES) loop of the PSI PMT for each audio stream carrying supplementary services. It may also be carried in this position for audio streams carrying main audio. The use of the descriptor is described in more detail in annex J.

The *mix_type* field in this descriptor can be used to indicate if the audio stream is a "complete and independent" stream, or a dependent stream which is not complete and independent and is intended to be mixed with a separate audio stream by the receiver, this is known as "receiver-mix audio", see clause J.2. The *mix_type* is also used to indicate when an SAOC-DE parametric data stream is embedded into an existing audio stream or carried as a separate component, see clause J.5.

The *editorial_classification* field can be used to indicate the editorial use of the audio stream. This can be used to indicate which streams are main programme audio that are targeted at the majority of viewers. The *editorial_classification* field can also be used to indicate the use of additional audio streams such as those for audiences with visual and/or hearing impairments. Examples include spoken subtitles, audio description or speech audio without music and sounds effects.

For receivers that do not have the capability to mix a dependent stream with another audio stream, a broadcaster can pre-mix the audio and broadcast it as a "complete and independent" additional audio stream, this is known as "broadcast-mix audio", see clause J.3.

The `supplementary_audio_descriptor` can override the ISO 639 [35] language code defined in the other descriptors in the same ES loop of the PMT, such as the `ISO_639_language_descriptor`. This is commonly used where the language code in the other descriptors (such as the `ISO_639_language_descriptor`) is set to aid legacy receivers. In such cases the more accurate language code is conveyed in the `supplementary_audio_descriptor`.

This descriptor shall be carried in the ES loop of the PSI PMT for each audio stream carrying a dependent stream. It may also be carried in this position for any audio stream. If a service contains a dependent audio stream this descriptor should be present in the ES loop of the PSI PMT carrying the audio streams with which the dependent stream could be mixed. The use of the descriptor is described in more detail in annex J.

NOTE: This descriptor cannot currently be used to describe audio which contains multiple audio sub-streams on a single PID.

For further rules on using this descriptor in the PMT, see clause M.2.

**Table 153: Supplementary audio descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| supplementary_audio_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    mix_type | 1 | uimsbf |
|    editorial_classification | 5 | uimsbf |
|    reserved_future_use | 1 | bslbf |
|    language_code_present | 1 | uimsbf |
|    if (language_code_present == 0b1) { | | |
|       ISO_639_language_code | 24 | bslbf |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       private_data_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the supplementary audio descriptor:

*mix_type***:** This 1-bit field indicates whether the audio stream is a complete and independent stream or whether it is a dependent stream intended to be mixed with a separate complete and independent audio stream according to ETSI TS 101 154 [14], annex G. It shall be coded according to table 154.

**Table 154: Mix type coding**

| *mix_type* (see note) | Description |
|---|---|
| 0 | the audio stream is a dependent stream and is intended to be mixed or combined with a separate complete and independent audio stream by the receiver |
| 1 | the audio stream is a complete and independent stream |
| NOTE: | Restrictions on valid combinations of *audio_type*, *mix_type*, and *editorial_classification* are given in clause J.4. |

*editorial_classification***:** This 5-bit field indicates the editorial intention of the audio stream according to table 155.

**Table 155: Editorial classification coding**

| *editorial_classification* (see note) | Description |
|---|---|
| 0x00 | main audio (contains all of the main audio components and can be presented on its own or mixed with a supplementary audio stream) This classification shall not be used for broadcast-mix audio (see clause J.3) e.g. audio streams that are premixed with visual impaired or hearing impaired audio. |
| 0x01 | audio description for the visually impaired (contains a spoken description of the visual content of the service) |
| 0x02 | clean audio for the hearing impaired |
| 0x03 | spoken subtitles for the visually impaired |
| 0x04 | dependent parametric data stream (not embedded) |
| 0x05 to 0x16 | reserved for future use |
| 0x17 | unspecific supplementary audio for the general audience |
| 0x18 to 0x1F | user defined |
| NOTE: | Restrictions on valid combinations of *audio_type*, *mix_type*, and *editorial_classification* are given in clause J.4. |

*language_code_present***:** When this 1-bit field is set to 0b1, it indicates that the *ISO_639_language_code* field is present and this *ISO_639_language_code* shall override other *ISO_639_language_code* contained in other descriptors in the same ES loop of the PMT. When set to 0b0, no language code field follows, and the language of the audio stream shall be indicated by another descriptor in the same ES loop of the PMT.

*ISO_639_language_code*: This 24-bit field identifies the language of the audio stream and overrides the value of any other *ISO_639_language_code* contained in other descriptors in the same ES loop. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*private_data_byte*: This is an 8-bit field, the value of which is privately defined.

## 6.4.12   Target region descriptor

The `target_region_descriptor` (see table 156) identifies a set of target regions. The set of target regions identified by the descriptor is the union of target region codes, identified by each loop in the descriptor, additionally qualified by the *country_code*. If the loop is empty, then this descriptor identifies a single country, by means of the *country_code*.

The `target_region_descriptor` is a scoping descriptor (see clause 6.5). The location of a particular instance of the `target_region_descriptor` defines the scope of the descriptor.

The scoping property is the set of target regions it identifies. The set of target regions identified at a particular scope is the union of the sets of target regions identified by all target region descriptors present at that scope in the same table.

The `target_region_descriptor` may appear:

1)   in the first loop of the NIT for all services within a network;

2)   in the TS loop of the NIT for all services within a DVB transport stream of a network;

3)   in the first loop of the BAT for all services within a bouquet;

4)   in the TS loop of the BAT for all services within a DVB transport stream of a bouquet;

5)   in the SDT for a single service within a DVB transport stream.

NOTE:   In revisions of the present document prior to 1.17.1, this ordered list used letters ("a)", "b)", and so on) to label list items. For technical reasons, starting with revision 1.17.1 list numbering now uses numbers as item labels ("1)", "2)", and so on). As the order of list items was not altered when introducing this change, references to letter-labelled list items can directly and unambiguously be converted to number-labelled list items by mapping the ordinal position of the letter-labelled item in the lexicographical ordering, to the same ordinal position in the numerical ordering of the number-labelled items (i.e. "a)" corresponds to "1)", "b)" corresponds to "2)", and so on).

Its use in other tables is not defined. This descriptor may appear multiple times in any descriptor loop of a table. The set of target regions defined in a descriptor loop of a table, is the union of the target regions defined by each `target_region_descriptor`.

The hierarchical level of the target region identified by a single loop of the `target_region_descriptor` depends on the value of the *region_depth* field:

* If the *region_depth* is 0, then the `target_region_descriptor` loop identifies a single country.

* If the value of the *region_depth* is 1, then the `target_region_descriptor` loop identifies a single primary region within a country.

* If the value of the *region_depth* is 2, then the `target_region_descriptor` loop identifies a single secondary region, within a primary region within a country.

* If the value of the *region_depth* is 3, the `target_region_descriptor` loop identifies a single tertiary region, within a secondary region within a primary region within a country.

**Table 156: Target region descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| target_region_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    country_code | 24 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       reserved_future_use | 5 | bslbf |
|       country_code_flag | 1 | bslbf |
|       region_depth | 2 | uimsbf |
|       if (country_code_flag == 0b1) { | | |
|          country_code | 24 | bslbf |
|       } | | |
|       if (region_depth >= 1) { | | |
|          primary_region_code | 8 | uimsbf |
|          if (region_depth >= 2) { | | |
|             secondary_region_code | 8 | uimsbf |
|             if (region_depth == 3) { | | |
|                tertiary_region_code | 16 | uimsbf |
|             } | | |
|          } | | |
|       } | | |
|    } | | |
| } | | |

Semantics for the target region descriptor:

**country_code:** This 24-bit field identifies a country or group of countries. It shall be coded as three characters, where each character is coded into 8-bits according to ISO/IEC 8859-1 [38] and inserted in order into the field. Individual countries shall be identified by using the alpha-3 code as specified in ISO 3166 [34]. A group of countries shall be identified by the string representation of a decimal number in the range 900 to 999. These numbers shall be coded according to ETSI TS 101 162 [15].

> EXAMPLE:     United Kingdom has 3-character code "GBR", which is coded as:
> 0b0100 0111 0100 0010 0101 0010.

The *country_code* can be signalled in two possible locations: the location outside the loop specifies the initial *country_code* which applies until overridden by a *country_code* defined inside the loop. A *country_code* defined inside the loop applies to all following loops in the descriptor until overridden by another *country_code* defined inside a subsequent loop.

**country_code_flag:** This 1-bit field if set to one indicates a *country_code* is defined inside the loop. The *country_code* applies to all following target regions in the descriptor until another *country_code* is defined inside a subsequent loop.

**region_depth:** This 2-bit field identifies the hierarchical depth of the target regions described by this descriptor loop. The semantics of the *region_depth* field are:

- If the *region_depth* is 0 then the target region is the entire country identified by the current value of *country_code*.

- If the *region_depth* is 1 then the target region is a primary region, identified by the *primary_region_code* within the identified country.

- If the *region_depth* is 2 then the target region is a secondary region, identified by the combination of the *secondary_region_code* and *primary_region_code* within the identified country.

- If the *region_depth* is 3 then the target region is a tertiary region, identified by the combination of the *tertiary_region_code*, *secondary_region_code*, and *primary_region_code* within the identified country.

**primary_region_code:** This 8-bit field identifies the target primary region within a country.

**secondary_region_code:** This 8-bit field identifies the target secondary region within a primary region within a country.

*tertiary_region_code*: This 16-bit field identifies the target tertiary region within a secondary region within a primary region within a country.

## 6.4.13   Target region name descriptor

A target region is a geographical area containing the intended audience of a broadcast. A target region can be either an entire country, or a geographical area within a country. A country may be subdivided into a set of primary regions and each primary region may be subdivided into a set of secondary regions and, finally, each secondary region may be subdivided into a set of tertiary regions. In general, target regions subdivide a country into a hierarchy of primary, secondary and tertiary regions.

The `target_region_name_descriptor` (see table 157) assigns a name to a target region. The target region name may be represented in multiple languages. For a particular target region within a country, only one name per language code shall be signalled.

This descriptor may only be present in the first loop of the BAT or in the first loop of the NIT. Multiple instances of this descriptor are allowed. If the name of a target region is defined in both the BAT and the NIT, then the name of that region in the BAT takes precedence.

All target regions, identified by a `target_region_descriptor` in a NIT, BAT, or SDT shall be named by a `target_region_name_descriptor`, defined in a NIT or a BAT. The hierarchical structure created may have variable depth. Broadcasters shall ensure an appropriate name is transmitted at each populated level and at all levels above them in the hierarchy.

**Table 157: Target region name descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `target_region_name_descriptor() {` | | |
| `    descriptor_tag` | 8 | uimsbf |
| `    descriptor_length` | 8 | uimsbf |
| `    descriptor_tag_extension` | 8 | uimsbf |
| `    country_code` | 24 | bslbf |
| `    ISO_639_language_code` | 24 | bslbf |
| `    for (i=0;i<N;i++) {` | | |
| `        region_depth` | 2 | uimsbf |
| `        name_length` | 6 | uimsbf |
| `        for (j=0;j<N;j++) {` | | |
| `            char` | 8 | uimsbf |
| `        }` | | |
| `        primary_region_code` | 8 | uimsbf |
| `        if (region_depth >= 2) {` | | |
| `            secondary_region_code` | 8 | uimsbf |
| `            if (region_depth == 3) {` | | |
| `                tertiary_region_code` | 16 | uimsbf |
| `            }` | | |
| `        }` | | |
| `    }` | | |
| `}` | | |

Semantics for the target region name descriptor:

*country_code*: This 24-bit field identifies a country or group of countries. It shall be coded as three characters, where each character is coded into 8-bits according to ISO/IEC 8859-1 [38] and inserted in order into the field. Individual countries shall be identified by using the alpha-3 code as specified in ISO 3166 [34]. A group of countries shall be identified by the string representation of a decimal number in the range 900 to 999. These numbers shall be coded according to ETSI TS 101 162 [15].

> EXAMPLE 1:    United Kingdom has 3-character code "GBR", which is coded as:
>                       0b0100 0111 0100 0010 0101 0010.

*ISO_639_language_code*: This 24-bit field identifies the language of the component text description which may be contained in this descriptor. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE 2: French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

*region_depth*: This 2-bit field identifies the hierarchical depth of the target regions described by this descriptor loop. The semantics of the *region_depth* field are:

- If the *region_depth* is 1 then the target region is a primary region, identified by the *primary_region_code* within the identified country.

- If the *region_depth* is 2 then the target region is a secondary region, identified by the combination of the *secondary_region_code* and *primary_region_code* within the identified country.

- If the *region_depth* is 3 then the target region is a tertiary region, identified by the combination of the *tertiary_region_code*, *secondary_region_code*, and *primary_region_code* within the identified country.

*name_length*: This 6-bit field specifies the length in bytes of the following target region name.

*char*: This is an 8-bit field, a sequence of which conveys the target region name. Text information is coded using the character sets and methods described in annex A.

*primary_region_code*: This 8-bit field identifies the target primary region within a country.

*secondary_region_code*: This 8-bit field identifies the target secondary region within a primary region within a country.

*tertiary_region_code*: This 16-bit field identifies the target tertiary region within a secondary region within a primary region within a country.

## 6.4.14 T2-MI descriptor

The T2MI_descriptor (see table 158) may be used in the PSI PMT to identify each PID carrying a single stream of DVB-T2-Modulator Interface (MI) data according to ETSI TS 102 773 [24] within a DVB transport stream.

**Table 158: T2-MI descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| T2MI_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    reserved_zero_future_use | 5 | bslbf |
|    t2mi_stream_id | 3 | uimsbf |
|    reserved_zero_future_use | 5 | bslbf |
|    num_t2mi_streams_minus_one | 3 | uimsbf |
|    reserved_zero_future_use | 7 | bslbf |
|    pcr_iscr_common_clock_flag | 1 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       reserved_zero_future_use | 8 | bslbf |
|    } | | |
| } | | |

Semantics for the T2-MI descriptor:

*t2mi_stream_id*: This 3-bit field identifies the DVB-T2-MI data stream being described. This field shall convey the same value as the *t2mi_stream_id* field in the header of the DVB-T2-MI packets defined in ETSI TS 102 773 [24], which are being carried in the elementary stream.

*num_t2mi_streams_minus_one*: This 3-bit field conveys the total number of DVB-T2-MI data streams required to generate the complete DVB-T2 signal. It shall be set to the same value for all DVB-T2-MI data streams that make up a composite signal.

EXAMPLE 1: Where only one DVB-T2-MI data stream is required by a modulator to generate a complete signal, the field has a value of 0.

EXAMPLE 2: Where a DVB-T2 composite signal is being generated that requires two DVB-T2-MI data streams to make the complete signal, the field has a value of 1.

*pcr_iscr_common_clock_flag***:** When this 1-bit field is set to 0b1, the Program Clock Reference (PCR) signalled in the PMT of the service carrying the DVB-T2-MI data, and the Input Stream Time Reference (ISCR) in the DVB-T2 Baseband Frame packets are generated from a common clock source (see annex G of ETSI TS 102 773 [24]).

## 6.4.15 URI linkage descriptor

The `URI_linkage_descriptor` (see table 159) identifies a resource obtainable via an Internet Protocol (IP) network. The usage of the resource identified in this way will depend upon both the location of this descriptor and the linkage type signalled in the descriptor.

**Table 159: URI linkage descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| URI_linkage_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    uri_linkage_type | 8 | uimsbf |
|    uri_length | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       uri_char | 8 | bslbf |
|    } | | |
|    if ((uri_linkage_type == 0x00) | | |
|    \|\| (uri_linkage_type == 0x01)) { | | |
|       min_polling_interval | 16 | uimsbf |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       private_data_byte | 8 | bslbf |
|    } | | |
| } | | |

Semantics for the URI linkage descriptor:

*uri_linkage_type***:** This is an 8-bit field specifying the type of Uniform Resource Identifier (URI) linkage e.g. to information. It shall be coded according to ETSI TS 101 162 [15].

*uri_length***:** This is an 8-bit field specifying the length of the following URI field.

*uri_char***:** This is an 8-bit field, a sequence of which conveys a URI string as defined in IETF RFC 3986 [33].

*min_polling_interval***:** This is a 16-bit field. The value of this field shall be the minimum time in intervals of two seconds between requests made by the receiver to this URI. A value of zero means that no minimum interval is implied. In this case receivers should choose a sensible default (for example a random value) or use a value defined by means outside of the scope of the present document (for example a value specified by local profile or platform).

> EXAMPLE 1: An interval of 10 minutes would be encoded as 300 (0x012C) since 300 intervals of 2 seconds each sum up to 600 seconds, which is equal to 10 minutes.

> EXAMPLE 2: An interval of 24 hours would be encoded as 43 200 (0xA8C0) since 43 200 intervals of 2 seconds each sum up to 86 400 seconds, which equals 24 hours.

> EXAMPLE 3: When the maximum value of 65 535 (0xFFFF) is encoded, the minimum interval is 131 070 seconds, which equals 36 hours, 24 minutes, and 30 seconds.

*private_data_byte***:** This is an 8-bit field, the value of which is privately defined.

## 6.4.16 Video depth range descriptor

### 6.4.16.1 Semantics and syntax of the video depth range descriptor

To assist receivers in optimizing the placement of graphics, like text or icons, they wish to display on top of plano-stereoscopic 3DTV video, the `video_depth_range_descriptor` (see table 160) indicates the intended depth range of the 3DTV video.

The `video_depth_range_descriptor` is a scoping descriptor (see clause 6.5). The location of a particular instance of the `video_depth_range_descriptor` defines the scope of the descriptor. If this descriptor occurs multiple times in a descriptor loop, then a particular *range_type* shall be present in only one instance of the `video_depth_range_descriptor`.

**Table 160: Video depth range descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| video_depth_range_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       range_type | 8 | uimsbf |
|       range_length | 8 | uimsbf |
|       if (range_type == 0x0) { | | |
|          production_disparity_hint_info() | | |
|       } else if (range_type == 0x1) { | | |
|          /* empty */ | | |
|       } else { | | |
|          for (j=0;j<N;j++) { | | |
|             range_selector_byte | 8 | bslbf |
|          } | | |
|       } | | |
|    } | | |
| } | | |

Semantics for the video depth range descriptor:

*range_type***:** This 8-bit field indicates the type of depth range described in this descriptor. It shall be coded according to table 161. A particular *range_type* shall occur in only one loop of this descriptor.

**Table 161: Range type coding**

| range_type | Description |
|---|---|
| 0x00 | production disparity hint |
| 0x01 | multi-region disparity Supplemental Enhancement Information (SEI) present |
| 0x02 to 0xFF | reserved for future use |

If the *range_type* is set to 0x01, it indicates the presence of multi-region disparity SEI data in the video elementary stream as defined in clause B.11 of ETSI TS 101 154 [14]. In this case, *range_length* shall be set to zero.

*range_length***:** This 8-bit field indicates the number of bytes following this field up to, but not including either the next *range_type* field or the end of the descriptor.

`production_disparity_hint_info`**():** This field shall be coded according to clause 6.4.16.2.

*range_selector_byte***:** For the purpose of application selection, the sequence of *range_selector_byte* is used. The definition of the *range_selector_byte* of the `video_depth_range_descriptor` will depend on the *range_type*.

### 6.4.16.2          Production disparity hint

When used, the `production_disparity_hint_info` conveys the recommended maximum and minimum disparity values as per the broadcaster's production guidelines used in the creation of the plano-stereoscopic 3DTV content, or accordingly modified by the broadcaster previous to transmission. Most of the content is expected to be most of the time within these disparity values, which shall be coded according to table 162. Disparity refers to the difference between the horizontal positions of a pixel in the left and right view that represents the same point in space. A positive value, right minus left, refers to a point away from the viewer and a negative value is one closer to the viewer.

**Table 162: Production disparity hint info**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `production_disparity_hint_info() {` | | |
| `    video_max_disparity_hint` | 12 | tcimsbf |
| `    video_min_disparity_hint` | 12 | tcimsbf |
| `}` | | |

Semantics for the production disparity hint info:

***video_max_disparity_hint***: This 12-bit field identifies the intended largest disparity according to the current production guidelines, which corresponds to an object at infinity, away from the viewer. If infinity disparity is unknown, then the disparity of the "furthest away object" should be given. This disparity is relative to a screen with a horizontal size of 11 520 pixels.

***video_min_disparity_hint***: This 12-bit field identifies the intended smallest disparity according to the current production guidelines, which corresponds to an object closest by the viewer. This disparity is relative to a screen with a horizontal size of 11 520 pixels.

## 6.4.17     VVC subpictures descriptor

The VVC subpictures descriptor is intended for use with VVC encoded personalization and/or accessibility services using VVC subpictures, such as described in ETSI TS 101 154 [14], annex M.

The descriptor shall be present in the PMT and may typically be present in the SDT or EIT for a service or event that relies on video from one or more video components encoded with subpictures. Presentation of the proper version of the service or event may require bitstream manipulation to extract and/or merge the appropriate VVC subpictures within one single bitstream representing the service or event.

The descriptor signals which video components and VVC subpictures are required for the service or event. The descriptor may also signal which kind of bitstream processing is required for reconstruction of the video bitstream (VVC subpictures extraction, merging or replacement). Since a VVC subpictures descriptor corresponds to one version of a service or event, several VVC subpictures descriptors typically need to be signalled in a service using VVC subpictures, one per version of the service.

Example guidelines about the usage of this descriptor for the example of mosaic or picture-in-picture services can be found in ETSI TS 101 211 [i.1].

**Table 162a: VVC subpictures descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| vvc_subpictures_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    default_service_mode | 1 | bslbf |
|    service_description_present | 1 | bslbf |
|    number_of_vvc_subpictures | 6 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       component_tag | 8 | uimsbf |
|       vvc_subpicture_id | 8 | uimsbf |
|    } | | |
|    reserved_zero_future_use | 5 | bslbf |
|    processing_mode | 3 | bslbf |
|    if (service_description_present == 0b1) { | | |
|       service_description_length | 8 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          char | 8 | bslbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the VVC subpictures descriptor

*default_service_mode:* This 1-bit field identifies the default VVC subpicture descriptor for presentation of the service. When this bit is set to 1, the VVC subpicture descriptor shall be considered as the default one to be selected by a receiver in the absence of user selection. When this bit is set to 0, the VVC subpicture descriptor describes an optional version of the service that may be selected by a user. Each PMT shall contain one and only one VVC subpicture descriptor with this field set to 1.

*service_description_present:* This 1-bit field specifies the presence of a text description for the service. A value of 0b1 specifies that a text description is present in the last loop of the descriptor.

*number_of_vvc_subpictures:* This 8-bit field specifies the number of VVC subpictures contained in the program and signalled in this descriptor. The minimum value of *number_of_vvc_subpictures* shall be 1.

*component_tag***:** This 8-bit field has the same value as the *component_tag* field in the stream identifier descriptor that shall be present in the PSI program map section for the video stream that contains the signalled VVC subpicture.

*vvc_subpicture_id:* This 8-bit field specifies the identifier of the VVC subpicture, as signalled in the slice header of the slices forming the VVC subpicture.

*processing_mode:* This 3-bit field indicates which kind of bitstream processing is required for reconstruction of the video bitstream for the program. It is encoded according to table 162b.

**Table 162b: Processing mode coding**

| *processing_mode* | Description |
|---|---|
| 0b000 | processing mode undefined |
| 0b001 | no bitstream processing necessary |
| 0b010 | merging of VVC subpictures into one bitstream necessary |
| 0b011 | reserved for future use |
| 0b100 | extraction of VVC subpictures from a bitstream necessary |
| 0b101 | reserved for future use |
| 0b110 | extraction and merging (replacement) of VVC subpictures necessary |
| 0b111 | reserved for future use |

*service_description_length:* This 8-bit field specifies the length in bytes of the following text describing the service.

*char:* This is an 8-bit field, a sequence of which conveys a text description of the service. Text information is coded using the character sets and methods described in annex A.

## 6.4.18   Service Prominence Descriptor

The service prominence descriptor offers a solution for compliance with Article 7a of Directive (EU) 2018/1808 [i.9] and allow signalling of Services of General Interest (SOGI). The descriptor may be used to list all SOGI within the transport stream descriptor loop in the NIT or the BAT, or to signal an individual service as a SOGI within the descriptor loop of the SDT for the service.

Therefore to fully identify the DVB triplet (transport_stream_id, original_network_id, service_id) of a service in the NIT or the BAT, the *service_id* is signalled within this descriptor and the *transport_stream_id* and the *original_network_id* are implied by the descriptor loop that this descriptor is placed in. Therefore to fully identify the DVB triplet (*transport_stream_id*, *original_network_id*, *service_id*) of a service in the SDT the three parts of the DVB triplet are implied by the descriptor loop that this descriptor is placed in.

Individual services that are broadcast to multiple geographical regions may have differing SOGI statuses or SOGI priorities within those regions. The service prominence descriptor allows signalling of a target region for each SOGI, and each SOGI may be defined multiple times with unique target region information.

**Table 162c: service_prominence_descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| service_prominence_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    SOGI_list_length | 8 | uimsbf |
|    if (SOGI_list_length > 0) { | | |
|       for (i=0;i<N;i++) { | | |
|          SOGI_flag | 1 | bslbf |
|          target_region_flag | 1 | bslbf |
|          service_flag | 1 | bslbf |
|          reserved_future_use | 1 | bslbf |
|          SOGI_priority | 12 | uimsbf |
|          if (service_flag == 0b1) { | | |
|             service_id | 16 | uimsbf |
|          } | | |
|          if (target_region_flag == 0b1) { | | |
|           target_region_loop_length | 8 | uimsbf |
|           for (j=0;j<N;j++) { | | |
|             reserved_future_use | 5 | bslbf |
|             country_code_flag | 1 | bslbf |
|             region_depth | 2 | uimsbf |
|             if (country_code_flag == 0b1) { | | |
|               country_code | 24 | bslbf |
|             } | | |
|             if (region_depth >= 1) { | | |
|               primary_region_code | 8 | uimsbf |
|               if (region_depth >= 2) { | | |
|                 secondary_region_code | 8 | uimsbf |
|                if (region_depth == 3) { | | |
|                  tertiary_region_code | 16 | uimsbf |
|                } | | |
|              } | | |
|             } | | |
|           } | | |
|          } | | |
|       } | | |
|    } | | |
|    for (i=0; i<N; i++) { | | |
|       private_data_byte | 8 | bslbf |
|    } | | |
| } | | |

Semantics for the service prominence descriptor:

***SOGI_list_length***: This 8-bit field indicates the total length in bytes of the following loop that describes the SOGI status of relevant services.

***SOGI_flag***: This 1-bit flag shall be set to 1 when the related service is allocated as a SOGI. For services that are not allocated as a SOGI, this flag shall be set to 0.

***target_region_flag***: This 1-bit flag shall be set to 1 when the descriptor contains further information about the region(s) relevant to SOGI signalling. Where the target region does not need to be further defined for SOGIs, this flag shall be set to 0. If this flag is set to 1 one or more target region descriptors shall be present.

***service_flag***: This 1-bit flag shall be set to 1 when the descriptor is used in the NIT or BAT. Where the descriptor is used in the SDT, this 1-bit flag shall be set to 0.

***SOGI_priority***: This 12-bit field indicates the priority of the SOGI. Where the service is not a SOGI, or has no SOGI priority defined, this field shall be set to the value of 0. A value of 1 is the highest SOGI priority and a value of 4 095 is the lowest SOGI priority.

***service_id***: This is a 16-bit field which uniquely identifies a service within a TS. The *service_id* is the same as the *program_number* in the corresponding *program_map_section*.

***target_region_loop_length***: This 8-bit field indicates the total length in bytes of the following loop that describes some or all of the target regions of the associated SOGI.

***country_code***: This 24-bit field identifies a country or group of countries. It shall be coded as three characters, where each character is coded into 8-bits according to ISO/IEC 8859-1 [38] and inserted in order into the field. Individual countries shall be identified by using the alpha-3 code as specified in ISO 3166 [34]. A group of countries shall be identified by the string representation of a decimal number in the range 900 to 999. These numbers shall be coded according to ETSI TS 101 162 [15].

> EXAMPLE:      United Kingdom has 3-character code "GBR", which is coded as:
> 0b0100 0111 0100 0010 0101 0010.

***country_code_flag***: This 1-bit field if set to 1 indicates a *country_code* is defined inside the loop. The country code applies to all following target regions in the descriptor until another country code is defined inside a subsequent loop.

***region_depth***: This 2-bit field identifies the hierarchical depth of the target region described by this descriptor loop. The semantics of the *region_depth* field are:

- If the *region_depth* is 0 then the target region is the entire country identified by the current value of *country_code*.

- If the *region_depth* is 1 then the target region is a primary region, identified by the *primary_region_code* within the identified country.

- If the *region_depth* is 2 then the target region is a secondary region, identified by the combination of the *secondary_region_code* and *primary_region_code* within the identified country.

- If the *region_depth* is 3 then the target region is a tertiary region, identified by the combination of the *tertiary_region_code*, *secondary_region_code* and *primary_region_code* within the identified country.

***primary_region_code***: This 8-bit field identifies the target primary region within a country.

***secondary_region_code***: This 8-bit field identifies the target secondary region within a primary region within a country.

***tertiary_region_code***: This 16-bit field identifies the target tertiary region within a secondary region within a primary region within a country.

# 6.5       Scoping rules for scoping descriptors

The information conveyed in some descriptors has a wider scope than the context in which the descriptor appears. Such descriptors are called scoping descriptors and the rules for their use are set forth in the present clause.

The location of a particular instance of a scoping descriptor defines the scope over which the scoping properties shall apply. These properties shall apply to all entities (e.g. services or events) within this scope unless overridden at a scope of greater precedence.

The scoping descriptor that holds for entities that fall outside of the scope of all explicit signalling is not defined by the present document.

In increasing precedence order, the descriptor loops where a scoping descriptor may appear, if allowed by the type of scoping descriptor, are:

1) NIT first loop **(lowest precedence)**

2) BAT first loop

3) NIT TS loop

4) BAT TS loop

5) SDT

6) EIT **(highest precedence)**

# 7 Storage Media Interoperability measures

## 7.0 Introduction

IEC 61883 [i.7] describes methods for delivering transport stream over the IEEE 1394.1 [i.8] to receivers. One likely source for this data is a digital storage device.

In certain cases DVB transport streams can be "incomplete", thus not conforming to the normal broadcast specifications. These "partial" TSs represent a subset of the data streams in the original TS. They may also be "discontinuous" - that is there may be changes in the TS or the subset of the TS presented and there may be temporal discontinuities. Clause 7 on Storage Media Interoperability (SMI) describes the SI and PSI required in the delivered data in these cases.

## 7.1 SMI tables

### 7.1.0 General principles

The SMI tables are encoded using the private section syntax defined in ISO/IEC 13818-1 [1]. The SIT may be up to 4 096 bytes long.

The bitstream presented at a digital interface shall either be a "complete" TS conforming to ETSI TS 101 154 [14] and with SI conforming to the present document, or it shall be a "partial" TS. In the latter case the SI and PSI shall conform to the present clause.

A "partial" TS shall not carry any SI tables other than the SIT and DIT described in clause 7.1.2 and clause 7.1.1 respectively. The PSI shall be restricted to the PAT and PMT instances required to correctly describe the streams within the "partial" TS.

The presence of the SIT in a bitstream identifies the bitstream as a "partial" TS coming from a digital interface. In this case the receiver should not expect the SI information required in a broadcast TS and should instead rely on that carried by the SIT.

The SIT contains a summary of all relevant SI information contained in the broadcast stream. The DIT shall be inserted at transition points where SI information is discontinuous. The use of the SIT and DIT is restricted to partial TS, they shall not be used in broadcasts.

## 7.1.1    Discontinuity Information Table

The DIT (see table 163) is to be inserted at transition points at which SI information may be discontinuous.

**Table 163: Discontinuity information section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| discontinuity_information_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    transition_flag | 1 | uimsbf |
|    reserved_future_use | 7 | bslbf |
| } | | |

Semantics for the discontinuity information section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b0.

*section_length*: This is a 12-bit field, which is set to 0x001.

*transition_flag*: This 1-bit flag indicates the kind of transition in the TS. When the bit is set to 0b1, it indicates that the transition is due to a change of the originating source. The change of the originating source can be a change of originating TS and/or a change of the position in the TS (e.g. in case of time-shift). When the bit is set to 0b0, it indicates that the transition is due to a change of the selection only, i.e. while staying within the same originating TS at the same position.

## 7.1.2    Selection Information Table

The SIT describes the services and events carried by the "partial" TS. SIT sections shall be coded according to table 164.

**Table 164: Selection information section**

| Syntax | Number of bits | Identifier |
|---|---|---|
| selection_information_section() { | | |
|    table_id | 8 | uimsbf |
|    section_syntax_indicator | 1 | bslbf |
|    reserved_future_use | 1 | bslbf |
|    reserved | 2 | bslbf |
|    section_length | 12 | uimsbf |
|    reserved_future_use | 16 | bslbf |
|    reserved | 2 | bslbf |
|    version_number | 5 | uimsbf |
|    current_next_indicator | 1 | bslbf |
|    section_number | 8 | uimsbf |
|    last_section_number | 8 | uimsbf |
|    reserved_future_use | 4 | bslbf |
|    transmission_info_descriptors_length | 12 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       descriptor() | | |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       service_id | 16 | uimsbf |
|       reserved_future_use | 1 | bslbf |
|       running_status | 3 | bslbf |
|       service_descriptors_length | 12 | uimsbf |
|       for (j=0;j<N;j++) { | | |
|          descriptor() | | |
|       } | | |
|    } | | |
|    CRC_32 | 32 | rpchof |
| } | | |

Semantics for the selection information section:

*table_id*: See table 2.

*section_syntax_indicator*: This 1-bit field shall be set to 0b1.

*section_length*: This is a 12-bit field. It specifies the number of bytes of the section, starting immediately following the *section_length* field and including the CRC. The value of the *section_length* field shall not exceed 4 093 so that the entire section has a maximum length of 4 096 bytes.

*version_number*: This 5-bit field is the version number of the sub_table. The *version_number* shall be incremented by 1 when a change in the information carried within the sub_table occurs. When it reaches value 31, it wraps around to 0. When the *current_next_indicator* is set to 0b1, then the *version_number* shall be that of the currently applicable sub_table. When the *current_next_indicator* is set to 0b0, then the *version_number* shall be that of the next applicable sub_table.

*current_next_indicator*: This 1-bit indicator, when set to 0b1 indicates that the sub_table is the currently applicable sub_table. When the bit is set to 0b0, it indicates that the sub_table sent is not yet applicable and shall be the next sub_table to be valid.

*section_number*: This 8-bit field gives the number of the section. The *section_number* shall be set to 0x00.

*last_section_number*: This 8-bit field specifies the number of the last section. The *last_section_number* shall be set to 0x00.

*transmission_info_descriptors_length*: This 12-bit field gives the total length in bytes of the following descriptor loop describing the transmission parameters of the partial TS.

*service_id*: This is a 16-bit field which serves as a label to identify this service from any other service within a TS. The *service_id* is the same as the *program_number* in the corresponding program_map_section, except that in the case of *service_type* = 0x04, 0x18, or 0x1B (NVOD reference services) the *service_id* does not have a corresponding *program_number*.

*running_status*: This is a 3-bit field indicating the status of the event in the original stream. This is the running status of the original present event. If no present event exists in the original stream the status is considered as "not running". The meaning of the *running_status* value is as defined in ETSI TS 101 211 [i.1].

**service_descriptors_length:** This 12-bit field gives the total length in bytes of the following descriptor loop containing SI related information on the service and event contained in the partial TS.

*CRC_32*: This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in annex A of ISO/IEC 13818-1 [1] after processing the entire section.

# 7.2 SMI descriptors

## 7.2.0 Introduction

This clause contains syntax and semantics for descriptors exclusively found in partial TSs.

## 7.2.1 Partial transport stream descriptor

The transmission information descriptor loop of the SIT contains all the information required for controlling and managing the play-out and copying of partial TS. The `partial_transport_stream_descriptor` (see table 165) is proposed to describe this information.

**Table 165: Partial transport stream descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| partial_transport_stream_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    reserved_future_use | 2 | bslbf |
|    peak_rate | 22 | uimsbf |
|    reserved_future_use | 2 | bslbf |
|    minimum_overall_smoothing_rate | 22 | uimsbf |
|    reserved_future_use | 2 | bslbf |
|    maximum_overall_smoothing_buffer | 14 | uimsbf |
| } | | |

Semantics for the partial transport stream descriptor:

*peak_rate*: The maximum momentary transport packet rate (i.e. 188 bytes divided by the time interval between start times of two succeeding TS packets). At least an upper bound for this *peak_rate* should be given. This 22-bit field is coded as a positive integer in units of 400 bit/s

*minimum_overall_smoothing_rate*: Minimum smoothing buffer leak rate for the overall TS (all packets are covered). This 22-bit field is coded as a positive integer in units of 400 bit/s. The value 0x3F FFFF is used to indicate that the minimum smoothing rate is undefined.

*maximum_overall_smoothing_buffer*: Maximum smoothing buffer size for the overall TS (all packets are covered). This 14-bit field is coded as a positive integer in units of 1 byte. The value 0x3FFF is used to indicate that the maximum smoothing buffer size is undefined.

# Annex A (normative):
# Coding of text characters

## A.0        General principles

Text items can optionally include information to select a wide range of character tables as indicated below.

If no character selection information is given in a text item, then the default character coding table (table 00 - Latin alphabet) of figure A.1 is assumed.

## A.1        Control codes

For one-byte character tables, the codes in the range 0x80 to 0x9F are assigned to control functions as shown in table A.1.

**Table A.1: Single byte control codes**

| Control code | UTF-8 encoded control code | Description |
|---|---|---|
| 0x80 to 0x85 | 0xC2 0x80 to 0xC2 0x85 | reserved for future use |
| 0x86 | 0xC2 0x86 | character emphasis on |
| 0x87 | 0xC2 0x87 | character emphasis off |
| 0x88 to 0x89 | 0xC2 0x88 to 0xC2 0x89 | reserved for future use |
| 0x8A | 0xC2 0x8A | Carriage Return/Line Feed (CR/LF) |
| 0x8B to 0x9F | 0xC2 0x8B to 0xC2 0x9F | user defined |

For two-byte character tables, the codes in the range 0xE080 to 0xE09F (which are within the private use area of ISO/IEC 10646 [52]) are assigned to control functions as shown in table A.2.

**Table A.2: Two byte control codes**

| Control code | UTF-8 encoded control code | Description |
|---|---|---|
| 0xE080 to 0xE085 | 0xEE 0x82 0x80 to 0xEE 0x82 0x85 | reserved for future use |
| 0xE086 | 0xEE 0x82 0x86 | character emphasis on |
| 0xE087 | 0xEE 0x82 0x87 | character emphasis off |
| 0xE088 to 0xE089 | 0xEE 0x82 0x88 to 0xEE 0x82 0x89 | reserved for future use |
| 0xE08A | 0xEE 0x82 0x8A | CR/LF |
| 0xE08B to 0xE09F | 0xEE 0x82 0x8B to 0xEE 0x82 0x9F | user defined |

## A.2        Selection of character table

Text fields can optionally start with non-spacing, non-displayed data which specifies the alternative character table to be used for the remainder of the text item.

If the first byte of the text field has a value in the range 0x20 to 0xFF then this and all subsequent bytes in the text item are coded using the default character coding table (table 00 - Latin alphabet) of figure A.1.

The selection of character table is indicated in table A.3.

**Table A.3: Character coding tables**

| First byte value | Character code table | Table description | Reproduced in |
|---|---|---|---|
| 0x01 | ISO/IEC 8859-5 [42] | Latin/Cyrillic alphabet | Figure A.2 |
| 0x02 | ISO/IEC 8859-6 [43] | Latin/Arabic alphabet | Figure A.3 |
| 0x03 | ISO/IEC 8859-7 [44] | Latin/Greek alphabet | Figure A.4 |
| 0x04 | ISO/IEC 8859-8 [45] | Latin/Hebrew alphabet | Figure A.5 |
| 0x05 | ISO/IEC 8859-9 [46] | Latin alphabet No. 5 | Figure A.6 |
| 0x06 | ISO/IEC 8859-10 [47] | Latin alphabet No. 6 | Figure A.7 |
| 0x07 | ISO/IEC 8859-11 [48] | Latin/Thai (draft only) | Figure A.8 |
| 0x08 | reserved for future use (see note) | | |
| 0x09 | ISO/IEC 8859-13 [49] | Latin alphabet No. 7 | Figure A.9 |
| 0x0A | ISO/IEC 8859-14 [50] | Latin alphabet No. 8 (Celtic) | Figure A.10 |
| 0x0B | ISO/IEC 8859-15 [51] | Latin alphabet No. 9 | Figure A.11 |
| 0x0C to 0x0F | reserved for future use | | |
| 0x10 | dynamically selected part of ISO/IEC 8859 [38] to [51] | See table A.4 | |
| 0x11 | ISO/IEC 10646 [52] | BMP | |
| 0x12 | KS X 1001-2014 [54] | Korean character set | |
| 0x13 | GB-2312-1980 [53] | Simplified Chinese character set | |
| 0x14 | Big5 subset of ISO/IEC 10646 [52] | Traditional Chinese character set | |
| 0x15 | UTF-8 encoding of ISO/IEC 10646 [52] | BMP | |
| 0x16 to 0x1E | reserved for future use | | |
| 0x1F | Described by *encoding_type_id* | Described by 8-bit *encoding_type_id* conveyed in second byte of the string | |
| NOTE: | The first byte value 0x08 was previously reserved for ISO/IEC 8859-12 which should have covered the Devanagri script. ISO/IEC however never published part 12 of the ISO/IEC 8859 series (see http://en.wikipedia.org/wiki/ISO/IEC_8859-12). Applications of the present document for the scripts used by South-Asian languages should use the Basic Multilingual Plane (BMP) of ISO/IEC 10646 [52], where appropriate glyphs are provided. | | |

If the first byte of the text field has value 0x1F then the following byte carries an 8-bit value (uimsbf) containing the *encoding_type_id*. This value indicates the encoding scheme of the string. It shall be coded according to ETSI TS 101 162 [15].

If the first byte of the text field has a value 0x10 then the following two bytes carry a 16-bit value (uimsbf) N to indicate that the remaining data of the text field is coded using the character code table specified in table A.4.

**Table A.4: Character Coding Tables for first byte 0x10**

| First byte value | Second byte value | Third byte value | Character code table | Table description | Reproduced in |
|---|---|---|---|---|---|
| 0x10 | 0x00 | 0x00 | reserved for future use | | |
| 0x10 | 0x00 | 0x01 | ISO/IEC 8859-1 [38] | West European | |
| 0x10 | 0x00 | 0x02 | ISO/IEC 8859-2 [39] | East European | |
| 0x10 | 0x00 | 0x03 | ISO/IEC 8859-3 [40] | South European | |
| 0x10 | 0x00 | 0x04 | ISO/IEC 8859-4 [41] | North and North-East European | |
| 0x10 | 0x00 | 0x05 | ISO/IEC 8859-5 [42] | Latin/Cyrillic | Figure A.2 |
| 0x10 | 0x00 | 0x06 | ISO/IEC 8859-6 [43] | Latin/Arabic | Figure A.3 |
| 0x10 | 0x00 | 0x07 | ISO/IEC 8859-7 [44] | Latin/Greek | Figure A.4 |
| 0x10 | 0x00 | 0x08 | ISO/IEC 8859-8 [45] | Latin/Hebrew | Figure A.5 |
| 0x10 | 0x00 | 0x09 | ISO/IEC 8859-9 [46] | West European & Turkish | Figure A.6 |
| 0x10 | 0x00 | 0x0A | ISO/IEC 8859-10 [47] | North European | Figure A.7 |
| 0x10 | 0x00 | 0x0B | ISO/IEC 8859-11 [48] | Thai | Figure A.8 |
| 0x10 | 0x00 | 0x0C | reserved for future use | | |
| 0x10 | 0x00 | 0x0D | ISO/IEC 8859-13 [49] | Baltic | Figure A.9 |
| 0x10 | 0x00 | 0x0E | ISO/IEC 8859-14 [50] | Celtic | Figure A.10 |
| 0x10 | 0x00 | 0x0F | ISO/IEC 8859-15 [51] | West European | Figure A.11 |
| 0x10 | 0x00 | 0x10 to 0xFF | reserved for future use | | |
| 0x10 | 0x01 to 0xFF | 0x00 to 0xFF | reserved for future use | | |

Figure A.1 through figure A.11 below use the following conventions:

- Colour coding is used to indicate the general glyph category:

| Colour | Description |
|---|---|
| light orange | letters of the Latin alphabet which are compatible with 7-bit US-ASCII encoding |
| light red | numbers of the Latin alphabet which are compatible with 7-bit US-ASCII encoding |
| light blue | marks, punctuation, symbols, and separators |
| light pink | non-spacing symbols (diacritical marks) |
| light green | region-specific alphabet symbols |

- Non-printing characters use the following mnemonics:

| Mnemonic | Description |
|---|---|
| SPC | space |
| NBSP | no-break space |
| SHY | soft hyphen |
| LRM | left-to-right mark |
| RLM | right-to-left mark |

**Character code table 00 - Latin alphabet with Unicode equivalents**

| First \ Second | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0- | | | | | | | | | | | | | | | | |
| 1- | | | | | | | | | | | | | | | | |
| 2- | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F |
| 3- | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F |
| 4- | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F |
| 5- | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F |
| 6- | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F |
| 7- | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | |
| 8- | | | | | | | | | | | | | | | | |
| 9- | | | | | | | | | | | | | | | | |
| A- | NBSP 00A0 | ¡ 00A1 | ¢ 00A2 | £ 00A3 | € 20AC | ¥ 00A5 | | § 00A7 | ¤ 00A4 | ' 2018 | " 201C | « 00AB | ← 2190 | ↑ 2191 | → 2192 | ↓ 2193 |
| B- | ° 00B0 | ± 00B1 | ² 00B2 | ³ 00B3 | × 00D7 | µ 00B5 | ¶ 00B6 | · 00B7 | ÷ 00F7 | ' 2019 | " 201D | » 00BB | ¼ 00BC | ½ 00BD | ¾ 00BE | ¿ 00BF |
| C- | | ` 0300 | ´ 0301 | ^ 0302 | ~ 0303 | ¯ 0304 | ˘ 0306 | · 0307 | ¨ 0308 | | ° 030A | ¸ 0327 | | ˝ 030B | ˛ 0328 | ˇ 030C |
| D- | — 2015 | ¹ 00B9 | ® 00AE | © 00A9 | TM 2122 | ♪ 266A | ¬ 00AC | ¦ 00A6 | | | | | ⅛ 215B | ⅜ 215C | ⅝ 215D | ⅞ 215E |
| E- | Ω 2126 | Æ 00C6 | Đ 0110 | ª 00AA | Ħ 0126 | | IJ 0132 | Ŀ 013F | Ł 0141 | Ø 00D8 | Œ 0152 | º 00BA | Þ 00DE | Ŧ 0166 | Ŋ 014A | 'n 0149 |
| F- | ĸ 0138 | æ 00E6 | đ 0111 | ð 00F0 | ħ 0127 | ı 0131 | ij 0133 | ŀ 0140 | ł 0142 | ø 00F8 | œ 0153 | ß 00DF | þ 00FE | ŧ 0167 | ŋ 014B | SHY 00AD |

NOTE: This table is a superset of ISO/IEC 6937 [37] with addition of the Euro symbol (U+20AC) in position 0xA4.

**Figure A.1: Character code table 00 - Latin alphabet with Unicode equivalents**

| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0- | | | | | | | | | | | | | | | | | 0- | |
| | 1- | | | | | | | | | | | | | | | | | 1- | |
| | 2- | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F | 2- | |
| | 3- | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F | 3- | |
| | 4- | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F | 4- | |
| | 5- | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F | 5- | |
| | 6- | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F | 6- | |
| | 7- | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | | 7- | |
| | 8- | | | | | | | | | | | | | | | | | 8- | |
| | 9- | | | | | | | | | | | | | | | | | 9- | |
| | A- | NBSP 00A0 | Ё 0401 | Ђ 0402 | Ѓ 0403 | Є 0404 | Ѕ 0405 | І 0406 | Ї 0407 | Ј 0408 | Љ 0409 | Њ 040A | Ћ 040B | Ќ 040C | SHY 00AD | Ў 040E | Џ 040F | A- | |
| | B- | А 0410 | Б 0411 | В 0412 | Г 0413 | Д 0414 | Е 0415 | Ж 0416 | З 0417 | И 0418 | Й 0419 | К 041A | Л 041B | М 041C | Н 041D | О 041E | П 041F | B- | |
| | C- | Р 0420 | С 0421 | Т 0422 | У 0423 | Ф 0424 | Х 0425 | Ц 0426 | Ч 0427 | Ш 0428 | Щ 0429 | Ъ 042A | Ы 042B | Ь 042C | Э 042D | Ю 042E | Я 042F | C- | |
| | D- | а 0430 | б 0431 | в 0432 | г 0433 | д 0434 | е 0435 | ж 0436 | з 0437 | и 0438 | й 0439 | к 043A | л 043B | м 043C | н 043D | о 043E | п 043F | D- | |
| | E- | р 0440 | с 0441 | т 0442 | у 0443 | ф 0444 | х 0445 | ц 0446 | ч 0447 | ш 0448 | щ 0449 | ъ 044A | ы 044B | ь 044C | э 044D | ю 044E | я 044F | E- | |
| | F- | № 2116 | ё 0451 | ђ 0452 | ѓ 0453 | є 0454 | ѕ 0455 | і 0456 | ї 0457 | ј 0458 | љ 0459 | њ 045A | ћ 045B | ќ 045C | § 00A7 | ў 045E | џ 045F | F- | |

NOTE 1:	For the Ruthenian language, the characters in code positions 0xA5 (S) and 0xF5 (s) are replaced by Ґ (U+0490) and ґ (U+0491), respectively.
NOTE 2:	This is compatible with ISO/IEC 8859-5 [42].

**Figure A.2: Character code table 01 - Latin/Cyrillic alphabet with Unicode equivalents**

| First \ Second | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0- | | | | | | | | | | | | | | | | |
| 1- | | | | | | | | | | | | | | | | |
| 2- | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F |
| 3- | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F |
| 4- | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F |
| 5- | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F |
| 6- | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F |
| 7- | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | |
| 8- | | | | | | | | | | | | | | | | |
| 9- | | | | | | | | | | | | | | | | |
| A- | NBSP 00A0 | | | | ¤ 00A4 | | | | | | | | ، 060C | SHY 00AD | | |
| B- | | | | | | | | | | | | ؛ 061B | | | | ؟ 061F |
| C- | | ء 0621 | آ 0622 | أ 0623 | ؤ 0624 | إ 0625 | ئ 0626 | ا 0627 | ب 0628 | ة 0629 | ت 062A | ث 062B | ج 062C | ح 062D | خ 062E | د 062F |
| D- | ذ 0630 | ر 0631 | ز 0632 | س 0633 | ش 0634 | ص 0635 | ض 0636 | ط 0637 | ظ 0638 | ع 0639 | غ 063A | | | | | |
| E- | ـ 0640 | ف 0641 | ق 0642 | ك 0643 | ل 0644 | م 0645 | ن 0646 | ه 0647 | و 0648 | ى 0649 | ي 064A | ً 064B | ُ 064C | ٌ 064D | َ 064E | ُ 064F |
| F- | ِ 0650 | ّ 0651 | ْ 0652 | | | | | | | | | | | | | |

NOTE: This is compatible with ISO/IEC 8859-6 [43].

**Figure A.3: Character code table 02 - Latin/Arabic alphabet with Unicode equivalents**

| First Nibble | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0-** | | | | | | | | | | | | | | | | |
| **1-** | | | | | | | | | | | | | | | | |
| **2-** | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F |
| **3-** | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F |
| **4-** | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F |
| **5-** | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F |
| **6-** | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F |
| **7-** | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | |
| **8-** | | | | | | | | | | | | | | | | |
| **9-** | | | | | | | | | | | | | | | | |
| **A-** | NBSP 00A0 | ' 2018 | ' 2019 | £ 00A3 | € 20AC | ₯ 20AF | ¦ 00A6 | § 00A7 | ¨ 00A8 | © 00A9 | ͺ 037A | « 00AB | ¬ 00AC | SHY 00AD | | ― 2015 |
| **B-** | ° 00B0 | ± 00B1 | ² 00B2 | ³ 00B3 | ΄ 0384 | ΅ 0385 | Ά 0386 | · 0387 | Έ 0388 | Ή 0389 | Ί 038A | » 0BB | Ό 038C | ½ 00BD | Ύ 038E | Ώ 038F |
| **C-** | ΐ 0390 | Α 0391 | Β 0392 | Γ 0393 | Δ 0394 | Ε 0395 | Ζ 0396 | Η 0397 | Θ 0398 | Ι 0399 | Κ 039A | Λ 039B | Μ 039C | Ν 039D | Ξ 039E | Ο 039F |
| **D-** | Π 03A0 | Ρ 03A1 | | Σ 03A3 | Τ 03A4 | Υ 03A5 | Φ 03A6 | Χ 30A7 | Ψ 03A8 | Ω 03A9 | Ϊ 03AA | Ϋ 03AB | ά 03AC | έ 03AD | ή 03AE | ί 03AF |
| **E-** | ΰ 03B0 | α 03B1 | β 03B2 | γ 03B3 | δ 03B4 | ε 03B5 | ζ 03B6 | η 03B7 | θ 03B8 | ι 03B9 | κ 03BA | λ 03BB | μ 03BC | ν 03BD | ξ 03BE | ο 03BF |
| **F-** | π 03C0 | ρ 03C1 | ς 03C2 | σ 03C3 | τ 03C4 | υ 03C5 | φ 03C6 | χ 03C7 | ψ 03C8 | ω 03C9 | ϊ 03CA | ϋ 03CB | ό 03CC | ύ 03CD | ώ 03CE | |

NOTE:     This is compatible with ISO/IEC 8859-7 [44].

**Figure A.4: Character code table 03 - Latin/Greek alphabet with Unicode equivalents**

| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Second Nibble | | | | | | | | | | | |
| 0- | | | | | | | | | | | | | | | | | | 0- |
| 1- | | | | | | | | | | | | | | | | | | 1- |
| 2- | | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F | 2- |
| 3- | | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F | 3- |
| 4- | | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F | 4- |
| 5- | | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F | 5- |
| 6- | | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F | 6- |
| 7- | | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | | 7- |
| 8- | | | | | | | | | | | | | | | | | | 8- |
| 9- | | | | | | | | | | | | | | | | | | 9- |
| A- | | NBSP 00A0 | | ¢ 00A2 | £ 00A3 | ¤ 00A4 | ¥ 00A5 | ¦ 00A6 | § 00A7 | ¨ 00A8 | © 00A9 | × 00D7 | « 00AB | ¬ 00AC | SHY 00AD | ® 00AE | ¯ 00AF | A- |
| B- | | ° 00B0 | ± 00B1 | 2 00B2 | 3 00B3 | ´ 00B4 | µ 00B5 | ¶ 00B6 | · 00B7 | ¸ 00B8 | 1 00B9 | ° 00F7 | » 00BB | ¼ 00BC | ½ 00BD | ¾ 00BE | | B- |
| C- | | | | | | | | | | | | | | | | | | C- |
| D- | | | | | | | | | | | | | | | | = 2017 | | D- |
| E- | | א 05D0 | ב 05D1 | ג 05D2 | ד 05D3 | ה 05D4 | ו 05D5 | ז 05D6 | ח 05D7 | ט 05D8 | י 05D9 | ך 05DA | כ 05DB | ל 05DC | ם 05DD | מ 05D# | ן 05DF | E- |
| F- | | נ 05E0 | ס 05E1 | ע 05E2 | ף 05E3 | פ 05E4 | ץ 05E5 | צ 05E6 | ק 05E7 | ר 05E8 | ש 05E9 | ת 05EA | | | LRM 200E | RLM 200F | | F- |
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | | |
| | | | | | | | | Second Nibble | | | | | | | | | | | |

NOTE:    This is compatible with ISO/IEC 8859-8 [45].

**Figure A.5: Character code table 04 - Latin/Hebrew alphabet with Unicode equivalents**

| | | Second Nibble | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F |
| 0- | | | | | | | | | | | | | | | | | |
| 1- | | | | | | | | | | | | | | | | | |
| 2- | | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F |
| 3- | | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F |
| 4- | | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F |
| 5- | | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F |
| 6- | | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F |
| 7- | | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | |
| 8- | | | | | | | | | | | | | | | | | |
| 9- | | | | | | | | | | | | | | | | | |
| A- | | NBSP 00A0 | ¡ 00A1 | ¢ 00A2 | £ 00A3 | ¤ 00A4 | ¥ 00A5 | ¦ 00A6 | § 00A7 | ¨ 00A8 | © 00A9 | × 00D7 | « 00AB | ¬ 00AC | SHY 00AD | ® 00AE | ‾ 00AF |
| B- | | ° 00B0 | ± 00B1 | ² 00B2 | ³ 00B3 | ´ 00B4 | µ 00B5 | ¶ 00B6 | · 00B7 | ¸ 00B8 | ¹ 00B9 | º 00F7 | » 00BB | ¼ 00BC | ½ 00BD | ¾ 00BE | ¿ 00BF |
| C- | | À 00C0 | Á 00C1 | Â 00C2 | Ã 00C3 | Ä 00C4 | Å 00C5 | Æ 00C6 | Ç 00C7 | È 00C8 | É 00C9 | Ê 00CA | Ë 00CB | Ì 00CC | Í 00CD | Î 00CE | Ï 00CF |
| D- | | Ğ 011E | Ñ 00D1 | Ò 00D2 | Ó 00D3 | Ô 00D4 | Õ 00D5 | Ö 00D6 | × 00D7 | Ø 00D8 | Ù 00D9 | Ú 00DA | Û 00DB | Ü 00DC | İ 0130 | Ş 015E | ß 00DF |
| E- | | à 00E0 | á 00E1 | â 00E2 | ã 00E3 | ä 00E4 | å 00E5 | æ 00E6 | ç 00E7 | è 00E8 | é 00E9 | ê 00EA | ë 00EB | ì 00EC | í 00ED | î 00EE | ï 00EF |
| F- | | ğ 011F | ñ 00F1 | ò 00F2 | ó 00F3 | ô 00F4 | õ 00F5 | ö 00F6 | ÷ 00F7 | ø 00F8 | ù 00F9 | ú 00FA | û 00FB | ü 00FC | ı 0131 | ş 015F | ÿ 00FF |
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F |
| | | Second Nibble | | | | | | | | | | | | | | | |

NOTE:     This is compatible with ISO/IEC 8859-9 [46].

**Figure A.6: Character code table 05 - Latin/Turkish alphabet with Unicode equivalents**

| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Second Nibble | | | | | | | | | | |
| First Nibble | 0- | | | | | | | | | | | | | | | | | 0- |
| | 1- | | | | | | | | | | | | | | | | | 1- |
| | 2- | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F | 2- |
| | 3- | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F | 3- |
| | 4- | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F | 4- |
| | 5- | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F | 5- |
| | 6- | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F | 6- |
| | 7- | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | | 007C | } 007D | ~ 007E | | 7- |
| | 8- | | | | | | | | | | | | | | | | | 8- |
| | 9- | | | | | | | | | | | | | | | | | 9- |
| | A- | NBSP 00A0 | Ą 0104 | Ē 0112 | Ģ 0122 | Ī 012A | Ĩ 0128 | Ķ 0136 | § 00A7 | Ļ 013B | Đ 0110 | Š 0160 | Ŧ 0166 | Ž 017D | SHY 00AD | Ū 016A | Ŋ 014A | A- |
| | B- | ° 00B0 | ą 0105 | ē 0113 | ġ 0123 | ī 012B | ĩ 0129 | ķ 0137 | · 00B7 | ļ 013C | đ 0111 | š 0161 | ŧ 0167 | ž 017E | — 2015 | ū 016B | ŋ 014B | B- |
| | C- | Ā 0100 | Á 00C1 | Â 00C2 | Ã 00C3 | Ä 00C4 | Å 00C5 | Æ 00C6 | Į 012E | Č 010C | É 00C9 | Ę 0118 | Ë 00CB | Ė 0116 | Í 00CD | Î 00CE | Ï 00CF | C- |
| | D- | Đ 00D0 | Ņ 0145 | Ō 014C | Ó 00D3 | Ô 00D4 | Õ 00D5 | Ö 00D6 | Ũ 0168 | Ø 00D8 | Ų 0172 | Ú 00DA | Û 00DB | Ü 00DC | Ý 00DD | Þ 00DE | ß 00DF | D- |
| | E- | ā 0101 | á 00E1 | â 00E2 | ã 00E3 | ä 00E4 | å 00E5 | æ 00E6 | į 012F | č 010D | é 00E9 | ę 0119 | ë 00EB | ė 0117 | í 00ED | î 00EE | ï 00EF | E- |
| | F- | ð 00F0 | ņ 0146 | ō 014D | ó 00F3 | ô 00F4 | õ 00F5 | ö 00F6 | ũ 0169 | ø 00F8 | ų 0173 | ú 00FA | û 00FB | ü 00FC | ý 00FD | þ 00FE | ĸ 0138 | F- |
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | |
| | | | | | | | | Second Nibble | | | | | | | | | | |

NOTE: This is compatible with ISO/IEC 8859-10 [47].

**Figure A.7: Character code table 06 - Latin/Nordic alphabet with Unicode equivalents**

| | | \-0 | \-1 | \-2 | \-3 | \-4 | \-5 | \-6 | \-7 | \-8 | \-9 | \-A | \-B | \-C | \-D | \-E | \-F | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | \-0 | \-1 | \-2 | \-3 | \-4 | \-5 | \-6 | \-7 | \-8 | \-9 | \-A | \-B | \-C | \-D | \-E | \-F | | |
| | 0- | | | | | | | | | | | | | | | | | 0- | |
| | 1- | | | | | | | | | | | | | | | | | 1- | |
| | 2- | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F | 2- | |
| | 3- | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F | 3- | |
| | 4- | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F | 4- | |
| | 5- | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F | 5- | |
| | 6- | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F | 6- | |
| | 7- | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | | 7- | |
| | 8- | | | | | | | | | | | | | | | | | 8- | |
| | 9- | | | | | | | | | | | | | | | | | 9- | |
| | A- | NBSP 00A0 | ก 0E01 | ข 0E02 | ฃ 0E03 | ค 0E04 | ฅ 0E05 | ฆ 0E06 | ง 0E07 | จ 0E08 | ฉ 0E09 | ช 0E0A | ซ 0E0B | ฌ 0E0C | ญ 0E0D | ฎ 0E0E | ฏ 0E0F | A- | |
| | B- | ฐ 0E10 | ฑ 0E11 | ฒ 0E12 | ณ 0E13 | ด 0E14 | ต 0E15 | ถ 0E16 | ท 0E17 | ธ 0E18 | น 0E19 | บ 0E1A | ป 0E1B | ผ 0E1C | ฝ 0E1D | พ 0E1E | ฟ 0E1F | B- | |
| | C- | ภ 0E20 | ม 0E21 | ย 0E22 | ร 0E23 | ฤ 0E24 | ล 0E25 | ฦ 0E26 | ว 0E27 | ศ 0E28 | ษ 0E29 | ส 0E2A | ห 0E2B | ฬ 0E2C | อ 0E2D | ฮ 0E2E | ฯ 0E2F | C- | |
| | D- | ะ 0E30 | ั 0E31 | า 0E32 | ำ 0E33 | ิ 0E34 | ี 0E35 | ึ 0E36 | ื 0E37 | ุ 0E38 | ู 0E39 | ฺ 0E3A | | | | | ฿ 0E3F | D- | |
| | E- | เ 0E40 | แ 0E41 | โ 0E42 | ใ 0E43 | ไ 0E44 | ๅ 0E45 | ๆ 0E46 | ็ 0E47 | ่ 0E48 | ้ 0E49 | ๊ 0E4A | ๋ 0E4B | ์ 0E4C | ํ 0E4D | ๎ 0E4E | ๏ 0E4F | E- | |
| | F- | ๐ 0E50 | ๑ 0E51 | ๒ 0E52 | ๓ 0E53 | ๔ 0E54 | ๕ 0E55 | ๖ 0E56 | ๗ 0E57 | ๘ 0E58 | ๙ 0E59 | ๚ 0E5A | ๛ 0E5B | | | | | F- | |
| | | \-0 | \-1 | \-2 | \-3 | \-4 | \-5 | \-6 | \-7 | \-8 | \-9 | \-A | \-B | \-C | \-D | \-E | \-F | | |

**Second Nibble** (top and bottom), **First Nibble** (left and right)

NOTE: This is compatible with ISO/IEC 8859-11 [48].

**Figure A.8: Character code table 07 - Latin/Thai alphabet with Unicode equivalents**

| First \ Second | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0- | | | | | | | | | | | | | | | | |
| 1- | | | | | | | | | | | | | | | | |
| 2- | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F |
| 3- | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F |
| 4- | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F |
| 5- | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F |
| 6- | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F |
| 7- | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | |
| 8- | | | | | | | | | | | | | | | | |
| 9- | | | | | | | | | | | | | | | | |
| A- | NBSP 00A0 | " 201D | ¢ 00A2 | £ 00A3 | ¤ 00A4 | „ 201E | ¦ 00A6 | § 00A7 | Ø 00D8 | © 00A9 | Ŗ 0156 | « 00AB | ¬ 00AC | SHY 00AD | ® 00AE | Æ 00C6 |
| B- | ° 00B0 | ± 00B1 | ² 00B2 | ³ 00B3 | " 201C | µ 00B5 | ¶ 00B6 | · 00B7 | ø 00F8 | ¹ 00B9 | ŗ 0157 | » 00BB | ¼ 00BC | ½ 00BD | ¾ 00BE | æ 00E6 |
| C- | Ą 0104 | Į 012E | Ā 0100 | Ć 0106 | Ä 00C4 | Å 00C5 | Ę 0118 | Ē 0112 | Č 010C | É 00C9 | Ź 0179 | Ė 0116 | Ģ 0122 | Ķ 0136 | Ī 012A | Ļ 013B |
| D- | Š 0160 | Ń 0143 | Ņ 0145 | Ó 00D3 | Ō 014C | Õ 00D5 | Ö 00D6 | × 00D7 | Ų 0172 | Ł 0141 | Ś 015A | Ū 016A | Ü 00DC | Ż 017B | Ž 017D | ß 00DF |
| E- | ą 0105 | į 012F | ā 0101 | ć 0107 | ä 00E4 | å 00E5 | ę 0119 | ē 0113 | č 010D | é 00E9 | ź 017A | ė 0117 | ģ 0123 | ķ 0137 | ī 012B | ļ 013C |
| F- | š 0161 | ń 0144 | ņ 0146 | ó 00F3 | ō 014D | õ 00F5 | ö 00F6 | ÷ 00F7 | ų 0173 | ł 0142 | ś 015B | ū 016B | ü 00FC | ż 017C | ž 017E | ' 2019 |

NOTE:    This is compatible with ISO/IEC 8859-13 [49].

**Figure A.9: Character code table 09 - Latin/Baltic Rim alphabet with Unicode equivalents**

| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Second Nibble | | | | | | | | | | |
| 0- | | | | | | | | | | | | | | | | | | 0- |
| 1- | | | | | | | | | | | | | | | | | | 1- |
| 2- | | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | - 002D | . 002E | / 002F | 2- |
| 3- | | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F | 3- |
| 4- | | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F | 4- |
| 5- | | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F | 5- |
| 6- | | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F | 6- |
| 7- | | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | | 7- |
| 8- | | | | | | | | | | | | | | | | | | 8- |
| 9- | | | | | | | | | | | | | | | | | | 9- |
| A- | | NBSP 00A0 | Ḃ 1E02 | ḃ 1E03 | £ 00A3 | Ċ 010A | ċ 010B | Ḋ 1E0A | § 00A7 | Ẁ 1E80 | © 00A9 | Ẃ 1E82 | ḋ 1E0B | Ỳ 1EF2 | SHY 00AD | ® 00AE | Ÿ 0178 | A- |
| B- | | Ḟ 1E1E | ḟ 1E1F | Ġ 0120 | ġ 0121 | Ṁ 1E40 | ṁ 1E41 | ¶ 00B6 | Ṗ 1E56 | ẁ 1E81 | ṗ 1E57 | ẃ 1E83 | Ṡ 1E60 | ỳ 1EF3 | Ẅ 1E84 | ẅ 1E85 | ṡ 1E61 | B- |
| C- | | À 00C0 | Á 00C1 | Â 00C2 | Ã 00C3 | Ä 00C4 | Å 00C5 | Æ 00C6 | Ç 00C7 | È 00C8 | É 00C9 | Ê 00CA | Ë 00CB | Ì 00CC | Í 00CD | Î 00CE | Ï 00CF | C- |
| D- | | Ŵ 0174 | Ñ 00D1 | Ò 00D2 | Ó 00D3 | Ô 00D4 | Õ 00D5 | Ö 00D6 | Ṫ 1E6A | Ø 00D8 | Ù 00D9 | Ú 00DA | Û 00DB | Ü 00DC | Ý 00DD | Ŷ 0176 | ß 00DF | D- |
| E- | | à 00E0 | á 00E1 | â 00E2 | ã 00E3 | ä 00E4 | å 00E5 | æ 00E6 | ç 00E7 | è 00E8 | é 00E9 | ê 00EA | ë 00EB | ì 00EC | í 00ED | î 00EE | ï 00EF | E- |
| F- | | ŵ 0175 | ñ 00F1 | ò 00F2 | ó 00F3 | ô 00F4 | õ 00F5 | ö 00F6 | ṫ 1E6B | ø 00F8 | ù 00F9 | ú 00FA | û 00FB | ü 00FC | ý 00FD | ŷ 0177 | ÿ 00FF | F- |
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | |
| | | | | | | | | Second Nibble | | | | | | | | | | |

NOTE:     This is compatible with ISO/IEC 8859-14 [50].

**Figure A.10: Character code table 0A - Latin/Celtic alphabet with Unicode equivalents**

| | | Second Nibble | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | |
| 0- | | | | | | | | | | | | | | | | | | 0- |
| 1- | | | | | | | | | | | | | | | | | | 1- |
| 2- | | SPC 0020 | ! 0021 | " 0022 | # 0023 | $ 0024 | % 0025 | & 0026 | ' 0027 | ( 0028 | ) 0029 | * 002A | + 002B | , 00C | – 002D | . 002E | / 002F | 2- |
| 3- | | 0 0030 | 1 0031 | 2 0032 | 3 0033 | 4 0034 | 5 0035 | 6 0036 | 7 0037 | 8 0038 | 9 0039 | : 003A | ; 003B | < 003C | = 003D | > 003E | ? 003F | 3- |
| 4- | | @ 0040 | A 0041 | B 0042 | C 0043 | D 0044 | E 0045 | F 0046 | G 0047 | H 0048 | I 0049 | J 004A | K 004B | L 004C | M 004D | N 004E | O 004F | 4- |
| 5- | | P 0050 | Q 0051 | R 0052 | S 0053 | T 0054 | U 0055 | V 0056 | W 0057 | X 0058 | Y 0059 | Z 005A | [ 005B | \ 005C | ] 005D | ^ 005E | _ 005F | 5- |
| 6- | | ` 0060 | a 0061 | b 0062 | c 0063 | d 0064 | e 0065 | f 0066 | g 0067 | h 0068 | i 0069 | j 006A | k 006B | l 006C | m 006D | n 006E | o 006F | 6- |
| 7- | | p 0070 | q 0071 | r 0072 | s 0073 | t 0074 | u 0075 | v 0076 | w 0077 | x 0078 | y 0079 | z 007A | { 007B | \| 007C | } 007D | ~ 007E | | 7- |
| 8- | | | | | | | | | | | | | | | | | | 8- |
| 9- | | | | | | | | | | | | | | | | | | 9- |
| A- | | NBSP 00A0 | ¡ 00A1 | ¢ 00A2 | £ 00A3 | € 20AC | ¥ 00A5 | Š 0160 | § 00A7 | š 0161 | © 00A9 | ª 00AA | « 00AB | ¬ 00AC | SHY 00AD | ® 00AE | ¯ 00AF | A- |
| B- | | ° 00B0 | ± 00B1 | ² 00B2 | ³ 00B3 | Ž 017D | µ 00B5 | ¶ 00B6 | · 00B7 | ž 017E | ¹ 00B9 | º 00BA | » 00BB | Œ 0152 | œ 0153 | Ÿ 0178 | ¿ 00BF | B- |
| C- | | À 00C0 | Á 00C1 | Â 00C2 | Ã 00C3 | Ä 00C4 | Å 00C5 | Æ 00C6 | Ç 00C7 | È 00C8 | É 00C9 | Ê 00CA | Ë 00CB | Ì 00CC | Í 00CD | Î 00CE | Ï 00CF | C- |
| D- | | Ð 00D0 | Ñ 00D1 | Ò 00D2 | Ó 00D3 | Ô 00D4 | Õ 00D5 | Ö 00D6 | × 00D7 | Ø 00D8 | Ù 00D9 | Ú 00DA | Û 00DB | Ü 00DC | Ý 00DD | Þ 00DE | ß 00DF | D- |
| E- | | à 00E0 | á 00E1 | â 00E2 | ã 00E3 | ä 00E4 | å 00E5 | æ 00E6 | ç 00E7 | è 00E8 | é 00E9 | ê 00EA | ë 00EB | ì 00EC | í 00ED | î 00EE | ï 00EF | E- |
| F- | | ð 00F0 | ñ 00F1 | ò 00F2 | ó 00F3 | ô 00F4 | õ 00F5 | ö 00F6 | ÷ 00F7 | ø 00F8 | ù 00F9 | ú 00FA | û 00FB | ü 00FC | ý 00FD | þ 00FE | ÿ 00FF | F- |
| | | -0 | -1 | -2 | -3 | -4 | -5 | -6 | -7 | -8 | -9 | -A | -B | -C | -D | -E | -F | |
| | | Second Nibble | | | | | | | | | | | | | | | | |

NOTE:	This is compatible with ISO/IEC 8859-15 [51].

**Figure A.11: Character code table 0B - Latin/Western European alphabet with Unicode equivalents**

# Annex B (informative):
# Void

# Annex C (informative): Conversion between time and date conventions

The types of conversion which may be required are summarized in figure C.1.
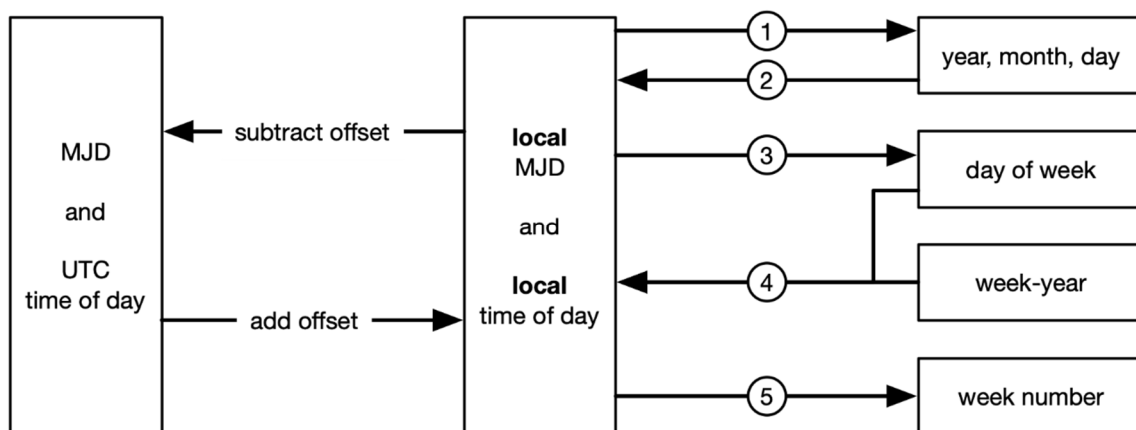


NOTE:     Offsets are positive for longitudes East of Greenwich and negative for longitudes West of Greenwich.

**Figure C.1: Conversion routes between MJD and UTC**

The conversion between MJD and UTC time of day, and the local MJD and local time of day is simply a matter of adding or subtracting the local offset. This process may, of course, involve a "carry" or "borrow" from the UTC affecting the MJD. The other five conversion routes shown on the diagram are detailed in the formulas below:

Symbols used:

| | |
|---|---|
| D | Day of month from 1 to 31 |
| int(…) | Integer part, ignoring remainder |
| $K, L, M', W', Y'$ | Intermediate variables |
| M | Month from January (1) to December (12) |
| MJD | Modified Julian Date |
| a mod b | Remainder (0 to b-1) after dividing integer a by integer b |
| UTC | Universal Time, Co-ordinated |
| WD | Day of week from Monday (1) to Sunday (7) |
| WN | Calender Week number according to ISO 8601-1:2019 [36]. Expected values in the range 1 to 53. |
| WY | "Week number" Year from 1900 |
| × | Multiplication |
| Y | Year from 1900 (e.g. for 2003, Y=103) |

1)    To find (Y,M,D) from MJD:

$$Y' = \text{int}\left(\frac{MJD - 15\,078,2}{365,25}\right)$$

$$M' = \text{int}\left(\frac{MJD - 14\,956,1 - \text{int}\left(Y' \times 365,25\right)}{30,6001}\right)$$

$$D = MJD - 14\,956 - \text{int}\left(Y' \times 365,25\right) - \text{int}\left(M' \times 30,6001\right)$$

$$\text{if } \left(\left(M' = 14\right) \text{ or } \left(M' = 15\right)\right), \text{ then } K = 1; \text{ else } K = 0$$

$$Y = Y' + K$$

$$M = M' - 1 - K \times 12$$

2)    To find MJD from (Y, M, D):

$$\text{if } \left((M = 1) \text{ or } (M = 2)\right), \text{ then } L = 1; \text{ else } L = 0$$

$$MJD = 14\,956 + D + \text{int}\left((Y - L) \times 365,25\right) + \text{int}\left((M + 1 + L \times 12) \times 30,6001\right)$$

3)    To find WD from MJD:

$$WD = \left((MJD + 2) \bmod 7\right) + 1$$

4)     To find MJD from (WY, WN, WD):

$$MJD = 15\,012 + WD + 7 \times \left(WN + \text{int}\left(\frac{WY \times 1\,461}{28} + 0,41\right)\right)$$

5)    To find (WY,WN) from MJD:

$$W = \text{int}\left(\frac{MJD}{7} - 2\,144,64\right)$$

$$WY = \text{int}\left(\frac{W \times 28}{1\,461} - 0,0079\right)$$

$$WN = W - \text{int}\left(\frac{WY \times 1\,461}{28} + 0,41\right)$$

NOTE 1:  In revisions of the present document prior to 1.17.1, this ordered list used letters ("a)", "b)", and so on) to label list items. For technical reasons, starting with revision 1.17.1 list numbering now uses numbers as item labels ("1)", "2)", and so on). As the order of list items was not altered when introducing this change, references to letter-labelled list items can directly and unambiguously be converted to number-labelled list items by mapping the ordinal position of the letter-labelled item in the lexicographical ordering, to the same ordinal position in the numerical ordering of the number-labelled items (i.e. "a)" corresponds to "1)", "b)" corresponds to "2)", and so on).

EXAMPLE:     MJD = 45 218
             Y = 82 (i.e. 1982)
             M = 9 (September)
             D = 6
             W = 4 315
             WY = 82 (i.e. 1982)
             WN = 36
             WD = 1 (Monday)

NOTE 2:  These formulas are applicable between the inclusive dates 1900 March 1 to 2100 February 28.

# Annex D (normative):
# Service information implementation of AC-3, Enhanced AC-3, and AC-4 audio in DVB systems

## D.0	Introduction

This annex describes the implementation and implementation guidelines for DVB service information for conveying AC-3, Enhanced AC-3, and AC-4 audio elementary streams within a DVB transport stream.

The transmission of AC-3, Enhanced AC-3, and AC-4 audio streams is optional in DVB systems, as specified in ETSI TS 101 154 [14]. However, if AC-3, Enhanced AC-3, or AC-4 audio streams are transmitted in a DVB system the specifications in this annex shall be followed.

## D.1	AC-3 and Enhanced AC-3 component types

Table D.1 shows the assignment of *component_type* values in the `component_descriptor` in the case that the *stream_content* value is set to 0x04, indicating the reference to an AC-3 or Enhanced AC-3 stream. If values of the *component_type* field other than those shown in table D.1 are conveyed, the *component_type* field may be ignored.

**Table D.1: AC-3 and Enhanced AC-3 component type value assignments**

| *component_type* bits | Description |
|---|---|
| $b_7$ | Enhanced AC-3 flag (see table D.2) |
| $b_6$ | Full service flag (see table D.3) |
| $b_5$ to $b_3$ | Service type flags (see table D.4) |
| $b_2$ to $b_0$ (see note) | Number of channels flags (see table D.5) |
| NOTE:	This bit is transmitted last (see clause 5.1.6). | |

**Table D.2: Enhanced AC-3 flag coding**

| Enhanced AC-3 flag $b_7$ | Description |
|---|---|
| 0b0 | Stream is AC-3 |
| 0b1 | Stream is Enhanced AC-3 |

**Table D.3: Full service flag coding**

| Full service flag $b_6$ | Description |
|---|---|
| 0b0 | Decoded audio stream is an associated service intended to be combined with another decoded audio stream before presentation to the listener |
| 0b1 | Decoded audio stream is a full service (suitable for decoding and presentation to the listener) |

**Table D.4: Service type flags coding**

| Service type flags | | | Description | Restrictions (see note 1) | |
|---|---|---|---|---|---|
| b$_5$ | b$_4$ | b$_3$ | | Full service flag (b$_6$) | Number of channels flags (b$_2$ to b$_0$ (see note 2)) |
| 0b0 | 0b0 | 0b0 | Complete Main (CM) | set to 0b1 | |
| 0b0 | 0b0 | 0b1 | Music and Effects (ME) | set to 0b0 | |
| 0b0 | 0b1 | 0b0 | Visually Impaired (VI) | | |
| 0b0 | 0b1 | 0b1 | Hearing Impaired (HI) | | |
| 0b1 | 0b0 | 0b0 | Dialogue (D) | set to 0b0 | |
| 0b1 | 0b0 | 0b1 | Commentary (C) | | set to 0b000 |
| 0b1 | 0b1 | 0b0 | Emergency (E) | set to 0b1 | set to 0b000 |
| 0b1 | 0b1 | 0b1 | Voice Over (VO) | set to 0b0 | set to 0b000 |
| 0b1 | 0b1 | 0b1 | Karaoke | set to 0b1 | set to 0b010, 0b011, or 0b100 |
| NOTE 1: The values of the service type flags are only valid if the conditions identified in the restrictions column are satisfied. | | | | | |
| NOTE 2: This bit is transmitted last (see clause 5.1.6). | | | | | |

**Table D.5: Number of channels flags coding**

| Number of channels flags | | | Description | Restrictions (see note 1) |
|---|---|---|---|---|
| b$_2$ | b$_1$ | b$_0$ (see note 2) | | Enhanced AC-3 flag (b$_7$) |
| 0b0 | 0b0 | 0b0 | Mono | |
| 0b0 | 0b0 | 0b1 | 1+1 Mode | |
| 0b0 | 0b1 | 0b0 | 2 channel (stereo) | |
| 0b0 | 0b1 | 0b1 | 2 channel Surround encoded (stereo) | |
| 0b1 | 0b0 | 0b0 | Multichannel audio (> 2 channels) | |
| 0b1 | 0b0 | 0b1 | Multichannel audio (> 5.1 channels) | set to 0b1 |
| 0b1 | 0b1 | 0b0 | Elementary stream contains multiple programmes carried in independent substreams | set to 0b1 |
| 0b1 | 0b1 | 0b1 | reserved for future use | |
| NOTE 1: The values of the number of channels flags are only valid if the conditions identified in the restrictions column are satisfied. | | | | |
| NOTE 2: This bit is transmitted last (see clause 5.1.6). | | | | |

# D.2     AC-3 descriptor

The `AC-3_descriptor` identifies an AC-3 audio elementary stream that has been coded in accordance with clause 6.2 of ETSI TS 101 154 [14]. The intended purpose is to provide configuration information for the IRD.

The descriptor shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop which describes an elementary stream carrying AC-3 audio coded in accordance with clause 6.2 of ETSI TS 101 154 [14].

The *descriptor_tag* provides a unique identification of the presence of the AC-3 elementary stream. Other optional fields in the descriptor may be used to provide identification of the component type mode of the AC-3 audio coded in the stream (*component_type* field) and indicate if the stream is a main AC-3 audio service (*mainid* field) or an associated AC-3 service (*asvc* field).

The descriptor has a minimum length of one byte, but may be longer depending upon the state of the flags and the *additional_info_byte* loop.

# D.3     AC-3 descriptor syntax and semantics

The `AC-3_descriptor` (see table D.6) is used in the PSI PMT to identify streams which carry AC-3 audio. The descriptor shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop.

**Table D.6: AC-3 descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| AC-3_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    component_type_flag | 1 | bslbf |
|    bsid_flag | 1 | bslbf |
|    mainid_flag | 1 | bslbf |
|    asvc_flag | 1 | bslbf |
|    reserved_flags | 4 | bslbf |
|    if (component_type_flag == 0b1) { | | |
|       component_type | 8 | uimsbf |
|    } | | |
|    if (bsid_flag == 0b1) { | | |
|       bsid | 8 | uimsbf |
|    } | | |
|    if (mainid_flag == 0b1) { | | |
|       mainid | 8 | uimsbf |
|    } | | |
|    if (asvc_flag == 0b1) { | | |
|       asvc | 8 | uimsbf |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       additional_info_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the AC-3 descriptor:

**component_type_flag:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *component_type* field in the descriptor.

**bsid_flag:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *bsid* field in the descriptor.

**mainid_flag:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *mainid* field in the descriptor.

**asvc_flag:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *asvc* field in the descriptor.

**reserved_flags:** These 1-bit fields are reserved for future use. They should always be set to 0b0.

**component_type:** This optional 8-bit field indicates the type of audio carried in the AC-3 elementary stream. It is set to the same value as the *component_type* field of the component_descriptor (see table D.1).

**bsid:** This optional 8-bit field indicates the AC-3 coding version. The three msb should always be set to 0b000. The five lsb are set to the same value as the *bsid* field in the AC-3 elementary stream, i.e. 0b0 1000 (= 8) or 0b0 0110 (= 6) in the current version of AC-3.

**mainid:** This optional 8-bit field identifies a main audio service and contains a number in the range 0 to 7 which identifies a main audio service. Each main service should be tagged with a unique number. This value is used as an identifier to link associated services with particular main services.

**asvc:** This 8-bit field is optional. Each bit (0 to 7) identifies with which main services this associated service is associated. The left most bit, bit 7, indicates whether this associated service may be reproduced along with main service number 7. If the bit has a value of 0b1, the service is associated with main service number 7. If the bit has a value of 0b0, the service is not associated with main service number 7.

**additional_info_byte:** These optional bytes are reserved for future use.

# D.4    Enhanced AC-3 descriptor

The enhanced_AC-3_descriptor identifies an Enhanced AC-3 audio elementary stream that has been coded in accordance with clause 6.2 of ETSI TS 101 154 [14]. The intended purpose is to provide configuration information for the IRD.

The descriptor shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop which describes an elementary stream carrying Enhanced AC-3 audio coded in accordance with clause 6.2 of ETSI TS 101 154 [14].

The *descriptor_tag* provides a unique identification of the presence of the Enhanced AC-3 elementary stream. Other optional fields in the descriptor may be used to provide identification of the *component_type* mode of the Enhanced AC-3 audio coded in the stream (*component_type* field) and indicate if the stream is a main Enhanced AC-3 audio service (*mainid* field) or an associated Enhanced AC-3 service (*asvc* field), or contains independent substreams (*substream1*, *substream2*, and *substream3* field).

The descriptor has a minimum length of one byte, but may be longer depending upon the state of the flags and the *additional_info_byte* loop.

# D.5     Enhanced AC-3 descriptor syntax and semantics

The `enhanced_AC-3_descriptor` (see table D.7) is used in the PSI PMT to identify streams which carry Enhanced AC-3 audio. The descriptor shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop.

**Table D.7: Enhanced AC-3 descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| enhanced_AC-3_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    component_type_flag | 1 | bslbf |
|    bsid_flag | 1 | bslbf |
|    mainid_flag | 1 | bslbf |
|    asvc_flag | 1 | bslbf |
|    mixinfoexists | 1 | bslbf |
|    substream1_flag | 1 | bslbf |
|    substream2_flag | 1 | bslbf |
|    substream3_flag | 1 | bslbf |
|    if (component_type_flag == 0b1) { | | |
|       component_type | 8 | uimsbf |
|    } | | |
|    if (bsid_flag == 0b1) { | | |
|       bsid | 8 | uimsbf |
|    } | | |
|    if (mainid_flag == 0b1) { | | |
|       mainid | 8 | uimsbf |
|    } | | |
|    if (asvc_flag == 0b1) { | | |
|       asvc | 8 | bslbf |
|    } | | |
|    if (substream1_flag == 0b1) { | | |
|       substream1 | 8 | uimsbf |
|    } | | |
|    if (substream2_flag == 0b1) { | | |
|       substream2 | 8 | uimsbf |
|    } | | |
|    if (substream3_flag == 0b1) { | | |
|       substream3 | 8 | uimsbf |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       additional_info_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the Enhanced AC-3 descriptor:

**component_type_flag**: This 1-bit field is mandatory. It should be set to 0b1 to include the optional *component_type* field in the descriptor.

*bsid_flag***:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *bsid* field in the descriptor.

*mainid_flag***:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *mainid* field in the descriptor.

*asvc_flag***:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *asvc* field in the descriptor.

*mixinfoexists***:** This 1-bit field is mandatory for Enhanced AC-3 streams. If set to 0b1 the Enhanced AC-3 stream contains metadata in independent substream 0 to control mixing with another AC-3 or Enhanced AC-3 stream.

*substream1_flag***:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *substream1* field in the descriptor. This flag should be set to 0b1 when the Enhanced AC-3 stream contains an additional programme carried in independent substream 1.

*substream2_flag***:** This 1-bit field is mandatory. It should be set to 0b1 to include the optional *substream2* field in the descriptor. This flag should be set to 0b1 when the Enhanced AC-3 stream contains an additional programme carried in independent substream 2.

*substream3_flag***:**This 1-bit field is mandatory. It should be set to 0b1 to include the optional *substream3* field in the descriptor. This flag should be set to 0b1 when the Enhanced AC-3 stream contains an additional programme carried in independent substream 3.

*component_type***:** This optional 8-bit field indicates the type of audio carried in the independent substream 0 of the Enhanced AC-3 elementary stream. It is set to the same value as the *component_type* field of the `component_descriptor` (see table D.1).

*bsid***:** This optional 8-bit field indicates the Enhanced AC-3 coding version. The three msb should always be set to 0b000. The five lsb are set to the same value as the *bsid* field in the Enhanced AC-3 elementary stream, i.e. 0b1 0000 (= 16) in the current version of Enhanced AC-3.

*mainid***:** This optional 8-bit field identifies a main audio service and contains a number in the range 0 to 7 which identifies a main audio service. Each main service should be tagged with a unique number. This value is used as an identifier to link associated services with particular main services.

*asvc***:** This 8-bit field is optional. Each bit (0 to 7) identifies with which main services this associated service is associated. The left most bit, bit 7, indicates whether this associated service may be reproduced along with main service number 7. If the bit has a value of 0b1, the service is associated with main service number 7. If the bit has a value of 0b0, the service is not associated with main service number 7.

*substream1***:** This optional 8-bit field indicates the type of audio carried in independent substream 1 of the Enhanced AC-3 elementary stream. The value assignments of each bit are indicated in table D.8.

*substream2***:** This optional 8-bit field indicates the type of audio carried in independent substream 2 of the Enhanced AC-3 elementary stream. The value assignments of each bit are indicated in table D.8.

*substream3***:** This optional 8-bit field indicates the type of audio carried in independent substream 3 of the Enhanced AC-3 elementary stream. The value assignments of each bit are indicated in table D.8.

*additional_info_byte***:** These optional bytes are reserved for future use.

**Table D.8: Substream 1 to 3 field bit value assignments**

| *substream1*, *substream2*, and *substream3* bits | Description |
|---|---|
| $b_7$ | Mixing metadata flag (see table D.9) |
| $b_6$ | Full service flag (see table D.3) |
| $b_5$ to $b_3$ | Service type flags (see table D.4) |
| $b_2$ to $b_0$ (see note) | Number of channels flags (see table D.10) |
| NOTE:     This bit is transmitted last (see clause 5.1.6). | |

If values of the *substream1*, *substream2*, or *substream3* fields other than those shown in table D.8 are conveyed, the respective substream field may be ignored.

**Table D.9: Mixing metadata flag coding**

| mixinfoexists | Description |
|---|---|
| 0b0 | No mixing metadata present in substream |
| 0b1 | Mixing metadata present in substream |

**Table D.10: Number of channels flags coding**

| Number of channels flags | | | Description |
|---|---|---|---|
| $b_2$ | $b_1$ | $b_0$ (see note) | |
| 0b0 | 0b0 | 0b0 | Mono |
| 0b0 | 0b0 | 0b1 | 1+1 Mode |
| 0b0 | 0b1 | 0b0 | 2 channel (stereo) |
| 0b0 | 0b1 | 0b1 | 2 channel Surround encoded (stereo) |
| 0b1 | 0b0 | 0b0 | Multichannel audio (> 2 channels) |
| 0b1 | 0b0 | 0b1 | Multichannel audio (> 5.1 channels) |
| 0b1 | 0b1 | 0b0 | reserved for future use |
| 0b1 | 0b1 | 0b1 | reserved for future use |
| NOTE: | This bit is transmitted last (see clause 5.1.6). | | |

# D.6    AC-4 descriptor

The AC-4_descriptor identifies an AC-4 audio elementary stream that has been coded in accordance with clauses 6.6 or 6.7 of ETSI TS 101 154 [14]. The intended purpose is to provide configuration information for the IRD.

The descriptor shall be included in a program_map_section at most once in each relevant *ES_info* descriptor loop which describes an elementary stream carrying AC-4 audio, coded in accordance with clause 6.6 or clause 6.7 of ETSI TS 101 154 [14].

The inclusion of the AC-4 Table Of Contents (TOC) is optional.

# D.7    AC-4 descriptor syntax and semantics

The AC-4_descriptor (see table D.11) uses the extension_descriptor syntax according to clause 6.2.16. The combination of *descriptor_tag* and *descriptor_tag_extension* provides a unique identification of the presence of an AC-4 elementary stream.

The AC-4_descriptor is used in the PSI PMT to identify streams which carry AC-4 audio. The descriptor shall be included in a program_map_section at most once in each relevant *ES_info* descriptor loop.

The minimum length of the AC-4_descriptor, as given by the *descriptor_length* field, is 2 bytes.

However, the descriptor may be longer depending upon the inclusion of optional fields, such as channel mode, the dialogue enhancement flag, the AC-4 TOC, and the *additional_info_byte* loop.

**Table D.11: AC-4 descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| AC-4_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    ac4_config_flag | 1 | bslbf |
|    ac4_toc_flag | 1 | bslbf |
|    reserved_zero_future_use | 6 | bslbf |
|    if (ac4_config_flag == 0b1) { | | |
|       ac4_dialog_enhancement_enabled | 1 | bslbf |
|       ac4_channel_mode | 2 | uimsbf |
|       reserved_zero_future_use | 5 | bslbf |
|    } | | |
|    if (ac4_toc_flag == 0b1) { | | |
|       ac4_toc_len | 8 | uimsbf |
|       for (i=0;i<N;i++) { | | |
|          ac4_dsi_byte | 8 | uimsbf |
|       } | | |
|    } | | |
|    for (i=0;i<N;i++) { | | |
|       additional_info_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the AC-4 descriptor:

**ac4_config_flag:** This 1-bit field indicates the presence of the *ac4_channel_mode* and the *ac4_dialog_enhancement_enabled* fields. This field shall be set to 0b1 if these optional fields are included in the descriptor.

**ac4_toc_flag:** This 1-bit field indicates the presence of the AC-4 TOC. This field shall be set to 0b1 if the optional AC-4 TOC is included in the descriptor.

**ac4_dialog_enhancement_enabled:** This 1-bit field shall be set to 0b1 if dialogue enhancement information is carried in the bit-stream, and to 0b0 otherwise. The *ac4_dialog_enhancement_enabled* field shall only be present if the *ac4_config_flag* is set to 0b1.

**ac4_channel_mode:** This field indicates the channel mode and shall be coded according to table D.12. The *ac4_channel_mode* field shall only be present if the *ac4_config_flag* is set to 0b1.

**Table D.12: AC-4 channel mode coding**

| *ac4_channel_mode* | Description |
|---|---|
| 0 | Mono content |
| 1 | Stereo content |
| 2 | Multichannel content |
| 3 | reserved for future use |

**ac4_toc_len:** This 8-bit field shall only be present if the *ac4_toc_flag* is set to 0b1. It shall indicate the length of the sequence of *ac4_dsi_byte* fields following it.

**ac4_dsi_byte:** This is an 8-bit field. The sequence of *ac4_dsi_byte* fields specifies the *ac4_dsi* field, which contains the AC-4 TOC (according to clause 6.6 of ETSI TS 101 154 [14]). The *ac4_dsi* field shall only be present if the *ac4_toc_len* is also present, and has a value greater than 0. The information contained in the *ac4_dsi* structure shall not be used to configure the AC-4 decoder. The AC-4 decoder shall obtain its configuration only from the AC-4 elementary stream.

**additional_info_byte:** These optional bytes are reserved for future use.

# D.8    Use of the supplementary audio descriptor with AC-4

When AC-4 as specified in clause 6.6 of ETSI TS 101 154 [14] is used to carry a service containing supplementary audio, the `supplementary_audio_descriptor` identifies the purpose of the associated audio service (see annex J). In this case, the contents of the `supplementary_audio_descriptor` should match the AC-4 TOC carried in the AC-4 elementary stream of the associated audio service.

# Annex E (normative):
# Usage of the scrambling descriptor

- Mixing of different scrambling modes within the same DVB transport stream:

  This situation may occur when a TS is made by multiplexing two or more independent TS streams.

- Mixing of different scrambling modes within the same service at the same time:

  This is not allowed. The same mode shall be used by all scrambled components of a service at the same time.

- Change of scrambling mode over time for a given service (e.g. from event to event):

  This situation may occur at any time, for instance when broadcasting events that were stored in scrambled mode or when inserting a local programme. Transitions should not be expected to be seamless.

# Annex F (informative):
# ISO 639 language descriptor for "original audio" soundtrack

Where audio in two languages is broadcast, ETSI TS 101 154 [14] specifies the order in which instances of the `ISO_639_language_descriptor` will be used in order to associate a language with its respective audio channel. In some cases, broadcasters may wish to signal that one of the channels contains the audio in the original language of the source, without specifying the exact language being used. This "original audio" language may be signalled with an *ISO_639_language_code* of "qaa", from the "local use" area of ISO 639-2 [35].

# Annex G (normative):
# Service information implementation of DTS coded audio in DVB systems

## G.0     Introduction

This annex describes the implementation and implementation guidelines for DVB service information for conveying DTS, DTS-HD, and DTS-UHD coded audio elementary streams within a DVB transport stream.

The transmission of DTS, DTS-HD, and DTS-UHD coded audio streams is optional in DVB systems, as specified in ETSI TS 101 154 [14]. However, if either DTS, DTS-HD, or DTS-UHD coded audio streams are transmitted in a DVB system the specifications in this annex shall be followed.

## G.1     DTS and DTS-HD descriptors

The DTS_descriptor identifies a DTS coded audio elementary stream that has been coded in accordance with clause 6.3 of ETSI TS 101 154 [14] for a core substream. If an extension substream is present, as described in clause 6.3 of ETSI TS 101 154 [14], then the DTS-HD_descriptor is used to identify the coded audio stream. In systems supporting the DTS-HD_descriptor, use of the DTS_descriptor is deprecated and the DTS-HD_descriptor is used to identify DTS and DTS-HD audio streams. The intended purpose is to provide configuration information for the IRD.

Either a DTS_descriptor or a DTS-HD_descriptor shall be included in a program_map_section in each relevant *ES_info* descriptor loop which describes an elementary stream containing DTS or DTS-HD coded audio coded in accordance with clause 6.3 of ETSI TS 101 154 [14].

The *descriptor_tag* provides a unique identification of the presence of either DTS or DTS-HD coded elementary streams. Other optional fields in the descriptor may be used to provide identification of the component type mode of the DTS or DTS-HD audio coded in the stream (*component_type* field).

## G.2     DTS descriptor

### G.2.0   Use of the DTS descriptor

The DTS_descriptor (see table G.1) is used in the PSI PMT to identify streams which carry DTS audio. The descriptor shall be included in a program_map_section at most once in each relevant *ES_info* descriptor loop. If a DTS_descriptor is present, a DTS-HD_descriptor shall not be present in the same descriptor loop.

### G.2.1   Syntax and semantics for the DTS descriptor

The DTS_descriptor is shown in table G.1. It is optional that the IRD decodes the DTS_descriptor.

**Table G.1: DTS descriptor**

| Syntax | Number of bits | Identifier |
|--------|:---:|:---:|
| DTS_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    sample_rate_code | 4 | bslbf |
|    bit_rate_code | 6 | bslbf |
|    nblks | 7 | bslbf |
|    fsize | 14 | uimsbf |
|    surround_mode | 6 | bslbf |
|    lfe_flag | 1 | uimsbf |
|    extended_surround_flag | 2 | uimsbf |
|    for (i=0;i<N;i++) { | | |
|       additional_info_byte | 8 | uimsbf |
|    } | | |
| } | | |

Semantics for the DTS descriptor:

*sample_rate_code*: This 4-bit field is equivalent to *SFREQ* in DTS Coherent Acoustics. Specification and details are listed in table G.2. While broadcasters may use only a subset of these, the complete table is given for consistency with the DTS Coherent Acoustics specification as defined in clause 6.3 of ETSI TS 101 154 [14].

**Table G.2: Sample rate code coding**

| *sample_rate_code* | Description |
|--------|--------|
| 0b0000 | invalid |
| 0b0001 | 8 kHz |
| 0b0010 | 16 kHz |
| 0b0011 | 32 kHz |
| 0b0100 | 64 kHz |
| 0b0101 | 128 kHz |
| 0b0110 | 11,025 kHz |
| 0b0111 | 22,05 kHz |
| 0b1000 | 44,1 kHz |
| 0b1001 | 88,02 kHz |
| 0b1010 | 176,4 kHz |
| 0b1011 | 12 kHz |
| 0b1100 | 24 kHz |
| 0b1101 | 48 kHz |
| 0b1110 | 96 kHz |
| 0b1111 | 192 kHz |

*bit_rate_code*: The specification and details of typical broadcast *bit_rate_code* applicable to the use of DTS audio in DVB transport streams are listed in table G.3. While broadcasters may use only a subset of these, the complete table of fixed transmission bit rate values is given for consistency with the DTS Coherent Acoustics specification as defined in clause 6.3 of ETSI TS 101 154 [14].

NOTE: It is recommended that DTS 5.1 compressed audio streams be transmitted at data rate of 384 kbit/s or above.

**Table G.3: Bit rate code coding**

| *bit_rate_code* (see note) | Description |
|---|---|
| 0bx0 0101 | 128 kbit/s |
| 0bx0 0110 | 192 kbit/s |
| 0bx0 0111 | 224 kbit/s |
| 0bx0 1000 | 256 kbit/s |
| 0bx0 1001 | 320 kbit/s |
| 0bx0 1010 | 384 kbit/s |
| 0bx0 1011 | 448 kbit/s |
| 0bx0 1100 | 512 kbit/s |
| 0bx0 1101 | 576 kbit/s |
| 0bx0 1110 | 640 kbit/s |
| 0bx0 1111 | 768 kbit/s |
| 0bx1 0000 | 960 kbit/s |
| 0bx1 0001 | 1 024 kbit/s |
| 0bx1 0010 | 1 152 kbit/s |
| 0bx1 0011 | 1 280 kbit/s |
| 0bx1 0100 | 1 344 kbit/s |
| 0bx1 0101 | 1 408 kbit/s |
| 0bx1 0110 | 1 411,2 kbit/s |
| 0bx1 0111 | 1 472 kbit/s |
| 0bx1 1000 | 1 536 kbit/s |
| 0bx1 1001 | 1 920 kbit/s |
| 0bx1 1010 | 2 048 kbit/s |
| 0bx1 1011 | 3 072 kbit/s |
| 0bx1 1100 | 3 840 kbit/s |
| 0bx1 1101 | open |
| 0bx1 1110 | variable |
| 0bx1 1111 | lossless |
| NOTE:     "x" indicates that the bit is reserved and should be ignored. | |

*nblks***:** This 7-bit field is equivalent to the *NBLKS* parameter. This equals the number of Pulse-Code Modulation (PCM) sample blocks. It indicates that there are $(NBLKS + 1)$ blocks (a block = 32 PCM samples per channel, corresponding to the number of PCM samples that are fed to the filterbank to generate one subband sample for each subband) in the current frame. The actual encoding window size is $32 \times (NBLKS + 1)$ PCM samples per channel. Valid range: 5 to 127. Invalid range: 0 to 4. For normal frames, this indicates a window size of either 2 048, 1 024 or 512 samples per channel. For termination frames, *NBLKS* can take any value in its valid range.

*fsize***:** This 14-bit field is equivalent to the *FSIZE* parameter. $(FSIZE + 1)$ is the byte size of the current primary audio frame. The valid range for *fsize* is 95 to 8 192. The invalid ranges for *fsize* are 0 to 94 and 8 193 to 16 384.

*surround_mode***:** This 6-bit field is equivalent to the *AMODE* parameter. The values for *surround_mode* applicable to the use of DTS audio in DVB transport streams are given in table G.4.

**Table G.4: Surround mode coding**

| surround_mode | Number of Channels / Channel Layout (see note) |
|---|---|
| 0b00 0000 | 1 / mono |
| 0b00 0010 | 2 / L + R (stereo) |
| 0b00 0011 | 2 / (L+R) + (L-R) (sum-difference) |
| 0b00 0100 | 2 / LT +RT (left and right total) |
| 0b00 0101 | 3 / C + L + R |
| 0b00 0110 | 3 / L + R+ S |
| 0b00 0111 | 4 / C + L + R+ S |
| 0b00 1000 | 4 / L + R+ SL+SR |
| 0b00 1001 | 5 / C + L + R+ SL+SR |
| 0b00 1010 | user defined |
| 0b00 1011 | user defined |
| 0b00 1100 | user defined |
| 0b00 1101 | user defined |
| 0b00 1110 | user defined |
| 0b00 1111 | user defined |
| 0b01 0000 to 0b11 1111 | user defined |
| NOTE:     L = left, R = right, C = centre, S = surround, T = total. | |

*lfe_flag*: The *lfe_flag* is set to 0 when the Low Frequency Effects (LFE) audio channel is off. The *lfe_flag* is set to 1 when the LFE audio channel is on.

*extended_surround_flag*: The *extended_surround_flag* indicates the presence of DTS extended surround rear centre audio as defined in clause 6.3 of ETSI TS 101 154 [14]. Its values are given in table G.5.

**Table G.5: Extended surround flag coding**

| extended_surround_flag | Description |
|---|---|
| 0b00 | no extended surround |
| 0b01 | matrixed extended surround |
| 0b10 | discrete extended surround |
| 0b11 | undefined |

*additional_info_byte*: This is an 8-bit field. The sequence of *additional_info_byte* fields specifies the *additional_info* field. It shall be coded according to clause 6.3 of ETSI TS 101 154 [14].

# G.3    DTS-HD descriptor

## G.3.1    DTS-HD descriptor syntax

Only one instance of the DTS-HD_descriptor shall be included in each relevant *ES_info* descriptor loop in the program_map_section. If a DTS-HD_descriptor is present, a DTS_descriptor shall not be present in the same descriptor loop.

Note that when both core and extension substreams are present, the core substream descriptor values are valid for the core as an independent asset in the event that an IRD can only decode the core substream.

The DTS-HD_descriptor is shown in table G.6. It is optional that the IRD decodes the DTS-HD_descriptor.

**Table G.6: DTS-HD descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `DTS-HD_descriptor() {` | | |
| `    descriptor_tag` | 8 | uimsbf |
| `    descriptor_length` | 8 | uimsbf |
| `    descriptor_tag_extension` | 8 | uimsbf |
| `    substream_core_flag` | 1 | bslbf |
| `    substream_0_flag` | 1 | bslbf |
| `    substream_1_flag` | 1 | bslbf |
| `    substream_2_flag` | 1 | bslbf |
| `    substream_3_flag` | 1 | bslbf |
| `    reserved_future_use` | 3 | bslbf |
| `    if (substream_core_flag == 0b1) {` | | |
| `        substream_info()` | | |
| `    }` | | |
| `    if (substream_0_flag == 0b1) {` | | |
| `        substream_info()` | | |
| `    }` | | |
| `    if (substream_1_flag == 0b1) {` | | |
| `        substream_info()` | | |
| `    }` | | |
| `    if (substream_2_flag == 0b1) {` | | |
| `        substream_info()` | | |
| `    }` | | |
| `    if (substream_3_flag == 0b1) {` | | |
| `        substream_info()` | | |
| `    }` | | |
| `    for (i=0;i<N;i++) {` | | |
| `        additional_info_byte` | 8 | uimsbf |
| `    }` | | |
| `}` | | |

Semantics for the DTS-HD descriptor:

*substream_core_flag***:** This 1-bit field is set to 0b1 if a core substream is present.

*substream_0_flag***:** This 1-bit field is set to 0b1 if an extension substream with nuExtSSIndex = 0 exists in the audio stream.

*substream_1_flag***:** This 1-bit field is set to 0b1 if an independent extension substream with nuExtSSIndex = 1 exists in the audio stream.

*substream_2_flag***:** This 1-bit field is set to 0b1 if an independent extension substream with nuExtSSIndex = 2 exists in the audio stream.

*substream_3_flag***:** This 1-bit field is set to 0b1 if an independent extension substream with nuExtSSIndex = 3 exists in the audio stream.

`substream_info():` This field is coded according to clause G.3.2.

*additional_info_byte***:** This 8-bit field is optional and reserves a place for future expansion of this descriptor.

## G.3.2   Substream information

The `substream_info` field is described in table G.7.

**Table G.7: Substream information**

| Syntax | Number of bits | Identifier |
|---|---|---|
| substream_info() { | | |
|    substream_length | 8 | uimsbf |
|    num_assets | 3 | uimsbf |
|    channel_count | 5 | uimsbf |
|    lfe_flag | 1 | bslbf |
|    sampling_frequency | 4 | uimsbf |
|    sample_resolution | 1 | bslbf |
|    reserved_future_use | 2 | bslbf |
|    for (i=0;i<N;i++) { | | |
|       asset_info() | | |
|    } | | |
| } | | |

Semantics for the substream information:

*substream_length***:** This 8-bit field specifies the total number of bytes following the *substream_length* field in the substream_info structure, including the embedded asset_info structures.

*num_assets***:** This 3-bit field represents the number of audio assets stored in the substream. The number of audio assets stored in the substream is equal to *num_assets* + 1. For a core substream, *num_assets* is always 0. For all independent extension substreams, this value is identical to *nuNumAssets* in the extension substream header.

*channel_count***:** This 5-bit field represents the maximum number of output channels including LFE (if present). Note that *channel_count* represents the maximum number of channels after all relevant assets are mixed together in multi-asset presentations and may be less than or equal to the number of channels of all assets combined.

*lfe_flag***:** This is a 1-bit field that is set to 0b1 if this substream contains an LFE channel.

*sampling_frequency***:** This 4-bit field represents the audio sampling frequency. Use table G.8 to determine the maximum sampling frequency stored in the elementary stream. Note that not all values in the table G.8 are applicable to core substreams).

**Table G.8: Sampling frequency coding**

| *sampling_frequency* | Description |
|---|---|
| 0 | 8 kHz |
| 1 | 16 kHz |
| 2 | 32 kHz |
| 3 | 64 kHz |
| 4 (see note) | 128 kHz |
| 5 | 22,05 kHz |
| 6 | 44,1 kHz |
| 7 | 88,2 kHz |
| 8 (see note) | 176,4 kHz |
| 9 (see note) | 352,8 kHz |
| 10 | 12 kHz |
| 11 | 24 kHz |
| 12 | 48 kHz |
| 13 | 96 kHz |
| 14 (see note) | 192 kHz |
| 15 (see note) | 348 kHz |
| NOTE:    This sampling frequency is not to be used with a core substream. | |

*sample_resolution***:** This 1-bit field is set to 0b1 if the decoded audio sample resolution is more than 16 bit. Otherwise, this field is set to 0.

asset_info()**:** This field is coded according to clause G.3.3. It appears as many times as indicated by the *num_assets* field.

# G.3.3 Asset information

The `asset_info()` field is described in table G.9.

**Table G.9: Asset information**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `asset_info() {` | | |
|    `asset_construction` | 5 | uimsbf |
|    `vbr_flag` | 1 | bslbf |
|    `post_encode_br_scaling_flag` | 1 | bslbf |
|    `component_type_flag` | 1 | bslbf |
|    `language_code_flag` | 1 | bslbf |
|    `if (post_encode_br_scaling_flag == 0b1) {` | | |
|       `bit_rate_scaled` | 13 | bslbf |
|    `} else {` | | |
|       `bit_rate` | 13 | uimsbf |
|    `}` | | |
|    `reserved_future_use` | 2 | bslbf |
|    `if (component_type_flag == 0b1) {` | | |
|       `component_type` | 8 | bslbf |
|    `}` | | |
|    `if (language_code_flag == 0b1) {` | | |
|       `ISO_639_language_code` | 24 | bslbf |
|    `}` | | |
| `}` | | |

Semantics for the asset information:

*asset_construction*: This 5-bit field provides details about the internal construction of the audio stream and is interpreted according to table G.10.

**Table G.10: Asset construction coding**

| *asset_construction* | Core substream | | | | Extension substream | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Core | XCH | X96 | XXCH | Core | XXCH | X96 | XBR | XLL | LBR |
| 1 | ✓ | - | - | - | - | - | - | - | - | - |
| 2 | ✓ | ✓ | - | - | - | - | - | - | - | - |
| 3 | ✓ | - | - | ✓ | - | - | - | - | - | - |
| 4 | ✓ | - | ✓ | - | - | - | - | - | - | - |
| 5 | ✓ | - | - | - | - | ✓ | - | - | - | - |
| 6 | ✓ | - | - | - | - | - | - | ✓ | - | - |
| 7 | ✓ | ✓ | - | - | - | - | - | ✓ | - | - |
| 8 | ✓ | - | - | ✓ | - | - | - | ✓ | - | - |
| 9 | ✓ | - | - | - | - | ✓ | - | ✓ | - | - |
| 10 | ✓ | - | - | - | - | - | ✓ | - | - | - |
| 11 | ✓ | ✓ | - | - | - | - | ✓ | - | - | - |
| 12 | ✓ | - | - | ✓ | - | - | ✓ | - | - | - |
| 13 | ✓ | - | - | - | - | ✓ | ✓ | - | - | - |
| 14 | ✓ | - | - | - | - | - | - | - | ✓ | - |
| 15 | ✓ | ✓ | - | - | - | - | - | - | ✓ | - |
| 16 | ✓ | - | ✓ | - | - | - | - | - | ✓ | - |
| 17 | - | - | - | - | - | - | - | - | ✓ | - |
| 18 | - | - | - | - | - | - | - | - | - | ✓ |
| 19 | - | - | - | - | ✓ | - | - | - | - | - |
| 20 | - | - | - | - | ✓ | ✓ | - | - | - | - |
| 21 | - | - | - | - | ✓ | - | - | - | ✓ | - |

*vbr_flag*: This 1-bit field is set to 0b1 only when the audio asset has a variable bit rate.

*post_encode_br_scaling_flag*: This 1-bit field is set to 0b1 if scaling of the bit stream has occurred after it was encoded, otherwise this field is set to 0b0.

*component_type_flag*: This 1-bit field is set to 0b1 if the field *component_type* is present.

*language_code_flag*: This 1-bit field is set to 0b1 when the *ISO_639_language_code* field is present.

*bit_rate_scaled*: This 13-bit field represents the scaled bit rate of the coded elementary stream as a 10,3 unsigned fractional fixed point value. This field is exists in the descriptor when the *post_encode_br_scaling_flag* is set to 0b1.

If the stream is variable bit rate, and dynamically updating the bit rate field is not possible or practical due to system limitations, or exceeds 8 191 kbit/s, then *bit_rate_scaled* is set to 0.

*bit_rate*: This 13-bit field is an unsigned integer representing the bit rate of the coded elementary stream in kbit/s, ranging from 1 kbit/s to 8 191 kbit/s. This field exists in the descriptor when the *post_encode_br_scaling_flag* is set to 0b0.

If the stream is variable bit rate, and dynamically updating the bit rate field is not possible or practical due to system limitations, then *bit_rate* is set to 0.

*component_type*: This field is coded according to clause G.3.4.

*ISO_639_language_code*: This 24-bit field identifies the language of the audio stream and overrides the value of any other *ISO_639_language_code* contained in other descriptors in the same ES loop. The *ISO_639_language_code* contains a 3-character code as specified by ISO 639-2 [35]. Both bibliographic (ISO 639-2 [35]/B) and terminological (ISO 639-2 [35]/T) codes from ISO 639-2 [35] may be used. Each character is coded into 8 bits according to ISO/IEC 8859-1 [38] and inserted in order into the 24-bit field.

EXAMPLE:      French has 3-character code "fre", which is coded as: 0b0110 0110 0111 0010 0110 0101.

# G.3.4   Component type

The *component_type* is an 8-bit field that specifies the type of the audio component as described in table G.11.

**Table G.11: Component type value assignments**

| *component_type* bits | Description |
|---|---|
| $b_7$ | reserved for future use |
| $b_6$ | Full service flag (see table G.12) |
| $b_5$ to $b_3$ | Service type flags (see table G.13) |
| $b_2$ to $b_0$ (see note) | Number of channels flags (see table G.14) |
| NOTE:       This bit is transmitted last (see clause 5.1.6). | |

**Table G.12: Full service flag coding**

| Full service flag $b_6$ | Description |
|---|---|
| 0b0 | Decoded audio stream is an associated service intended to be combined with another decoded audio stream before presentation to the listener |
| 0b1 | Decoded audio stream is a full service (suitable for decoding and presentation to the listener) |

**Table G.13: Service type flags coding**

| Service type flags | | | Description | Restrictions (see note 1) | |
|---|---|---|---|---|---|
| $b_5$ | $b_4$ | $b_3$ | | Full service flag ($b_6$) | Number of channels flags ($b_2$ to $b_0$ (see note 2)) |
| 0b0 | 0b0 | 0b0 | Complete Main (CM) | set to 0b1 | |
| 0b0 | 0b0 | 0b1 | Music and Effects (ME) | set to 0b0 | |
| 0b0 | 0b1 | 0b0 | Visually Impaired (VI) | | |
| 0b0 | 0b1 | 0b1 | Hearing Impaired (HI) | | |
| 0b1 | 0b0 | 0b0 | Dialogue (D) | set to 0b0 | |
| 0b1 | 0b0 | 0b1 | Commentary (C) | | set to 0b000 |
| 0b1 | 0b1 | 0b0 | Emergency (E) | set to 0b1 | set to 0b000 |
| 0b1 | 0b1 | 0b1 | Voice Over (VO) | set to 0b0 | set to 0b000 |
| 0b1 | 0b1 | 0b1 | reserved for future use | set to 0b1 | |
| NOTE 1: The values of the service type flags are only valid if the conditions identified in the restrictions column are satisfied. | | | | | |
| NOTE 2: This bit is transmitted last (see clause 5.1.6). | | | | | |

**Table G.14: Number of channels flags coding**

| Number of channels flags | | | Description |
|---|---|---|---|
| $b_2$ | $b_1$ | $b_0$ (see note) | |
| 0b0 | 0b0 | 0b0 | Mono |
| 0b0 | 0b0 | 0b1 | reserved for future use |
| 0b0 | 0b1 | 0b0 | 2 channel (stereo, LoRo) |
| 0b0 | 0b1 | 0b1 | 2 channel matrix encoded (stereo, LtRt) |
| 0b1 | 0b0 | 0b0 | Multichannel audio (> 2 channels) |
| 0b1 | 0b0 | 0b1 | reserved for future use |
| 0b1 | 0b1 | 0b0 | reserved for future use |
| 0b1 | 0b1 | 0b1 | reserved for future use |
| NOTE: This bit is transmitted last (see clause 5.1.6). | | | |

# G.4 Use of DTS-HD in Receiver Mixed Applications for Single PID and Multiple PID Implementations

For applications where audio is being mixed in the receiver, DTS-HD can accommodate two methods of asset delivery:

- Multiple PID delivery, where each PID is delivering a single asset.

  For multiple PID delivery, the DTS-HD stream contains a single asset and a `component_descriptor` in the PMT loop. In this case, *component_type_flag* and *language_code_flag* may be set to 0, thus disabling the respective fields. If the respective flags are enabled, the corresponding values need to agree with the `component_descriptor` in the PMT loop.

- Single PID delivery, where all assets are combined into a single multi-asset DTS-HD audio stream.

  For single PID delivery where multiple assets are delivered in a single DTS-HD bitstream, multiple `asset_info` components exist in the `DTS-HD_descriptor`. In this case, the *component_type_flag* and *language_code_flag* are normally set, thus enabling the respective fields for each asset. The receiver can select which program combination is to be mixed and presented. The `component_descriptor` in the PMT loop describes the default complete main presentation as configured by the broadcaster.

# G.5    DTS-UHD descriptors

## G.5.1    DTS-UHD descriptor

The DTS-UHD_descriptor (see table G.15) uses the extension_descriptor syntax according to clause 6.2.16.

The DTS-UHD_descriptor is used in the PSI PMT to identify streams which carry DTS-UHD audio. At most one instance of the DTS-UHD_descriptor shall be included in each relevant *ES_info* descriptor loop.

**Table G.15: DTS-UHD descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| DTS-UHD_descriptor() { |  |  |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    descriptor_tag_extension | 8 | uimsbf |
|    DecoderProfileCode | 6 | uimsbf |
|    FrameDurationCode | 2 | uimsbf |
|    MaxPayloadCode | 3 | uimsbf |
|    DTS_reserved | 2 | bslbf |
|    StreamIndex | 3 | uimsbf |
|    for (i=0;i<N;i++) { |  |  |
|       codec_selector_byte | 8 | uimsbf |
|    } |  |  |
| } |  |  |

Semantics for the DTS-UHD descriptor:

*DecoderProfileCode***:** This 6-bit field indicates the DTS-UHD decoder profile required to decode this stream. The relationship between the *DecoderProfile* and *DecoderProfileCode* shall be as follows: *DecoderProfile* = *DecoderProfileCode* + 2.

*DecoderProfileCode* shall be either 0, indicating DTS-UHD Profile 2, or 1 indicating DTS-UHD Profile 3. Other values shall be reserved.

*FrameDurationCode***:** This 2-bit field indicates *FrameDuration* in units of audio PCM samples at the base sampling frequency. It shall be coded according to table G.16. The frame duration in seconds is *FrameDuration* divided by the clock rate, which is always 48 kHz.

**Table G.16: FrameDurationCode coding**

| *FrameDurationCode* | Description |
|---|---|
| 0 | 512 samples |
| 1 | 1 024 samples |
| 2 | 2 048 samples |
| 3 | 4 096 samples |

*MaxPayloadCode***:** This 3-bit field indicates *MaxPayload*, the maximum size of the audio payload. It shall be coded according to table G.17.

**Table G.17: MaxPayloadCode coding**

| MaxPayloadCode | Description |
|---|---|
| 0 | 2 048 byte |
| 1 | 4 096 byte |
| 2 | 8 192 byte |
| 3 | 16 384 byte |
| 4 | 32 768 byte |
| 5 | 65 536 byte |
| 6 | 131 072 byte |
| 7 | reserved for future use |

NOTE 1: *MaxPayload* is not the size of the largest audio frame in the presentation, but rather a "not to exceed" value for buffer configuration and digital audio interface purposes, and is inclusive of all required preambles, headers, burst spacing, etc. It can for example be used to determine the minimum frame rate required to transfer the DTS-UHD stream using a digital audio interface.

***DTS_reserved***: This 2-bit field should be set to 0b00 for DVB applications. If this field is not 0b00, then a codec selector field will be present.

***StreamIndex***: This 3-bit field identifies the stream priority for prioritizing mixing metadata when an audio program is delivered using multiple streams. The main stream shall be assigned *StreamIndex* value 0, while auxiliary streams shall be assigned *StreamIndex* values in the range of 1 to 7.

If the audio program is delivered in a single stream, *StreamIndex* shall be set to 0.

***codec_selector_byte***: This is an 8-bit field. The sequence of *codec_selector_byte* fields specifies the codec selector field. The syntax and semantics of the codec selector field shall be defined by the respective codec. This field shall be ignored by receivers.

NOTE 2: The length of the codec selector field is implied by the overall length of the `DTS-UHD_descriptor`.

EXAMPLE: When the *descriptor_length* field has a value of 5, then the codec selector field has a length of $5\text{byte} - 3\text{byte} = 2\text{byte}$.

# G.5.2 DTS-UHD and the audio preselection descriptor

## G.5.2.1 The DTS-UHD Broadcast Chunk and audio preselections

When the DTS-UHD *BroadcastChunk* (defined in ETSI TS 101 154 [14]) is present in the main DTS-UHD stream, the corresponding parameters in the `audio_preselection_descriptor` can be determined from the *BroadcastChunk*, as described below.

***num_preselections***: This is the number of audio preselections being defined in this descriptor. At least one audio preselection is required. If all audio preselections defined in the *BroadcastChunk* are defined in the `audio_preselection_descriptor`, then the total number of audio preselections can be calculated as:

$$\text{num\_preselections} = \sum_{i=0}^{\text{numLanguages}} (\text{numSelectionSets}[i]+1)$$

***preselection_id***: *preselection_id* identifies a particular DTS-UHD Program as follows:

$$\text{preselection\_id} = \text{ProgramIndex} + 1 + \begin{cases} 0 & \text{when LanguageIndex} = 0 \\ \sum_{i=0}^{\text{LanguageIndex}-1} (\text{numSelectionSets}[i]+1) & \text{when LanguageIndex} > 0 \end{cases}$$

where *LanguageIndex* is a zero-based index into the language table represented by *i* in table 28 in ETSI TS 101 154 [14], and *ProgramIndex* is the zero-based index into a given language group represented by *j* in table 28 in ETSI TS 101 154 [14].

## G.6     Use of the supplementary audio descriptor with DTS-UHD

When DTS-UHD, as specified in clause 6.9 of ETSI TS 101 154 [14] and not including the *BroadcastChunk*, and an associated `audio_preselection_descriptor`, is used to carry a service containing supplementary audio, the `supplementary_audio_descriptor` identifies the purpose of the associated audio service (see annex J). In this case, the contents of the `supplementary_audio_descriptor` should match the content encoded in the elementary stream.

# Annex H (normative):
# Service information implementation of AAC coded audio in DVB systems

## H.0    Introduction

Annex H describes the implementation and implementation guidelines for DVB service information for conveying Advanced Audio Coding (AAC) coded audio elementary streams within a DVB transport stream.

The transmission of AAC coded audio streams is optional in DVB systems, as specified in ETSI TS 101 154 [14]. However, if AAC coded audio streams are transmitted in a DVB system the specifications in annex H shall be followed.

## H.1    AAC Audio descriptor

The `AAC_descriptor` identifies an AAC coded audio elementary stream that has been coded in accordance with clause 6.4 of ETSI TS 101 154 [14]. The intended purpose is to provide configuration information for the IRD.

The `AAC_descriptor` shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop which describes an elementary stream containing AAC coded audio coded in accordance with clause 6.4 of ETSI TS 101 154 [14].

The *descriptor_tag* of the `AAC_descriptor` provides a unique identification of the presence of the AAC coded elementary stream. Other optional fields in the `AAC_descriptor` may be used to provide identification of the *component_type* mode of the AAC audio coded in the stream (*AAC_type* field).

## H.2    AAC descriptor

### H.2.0    Use of the AAC descriptor

The `AAC_descriptor` (see table H.1) is used in the PSI PMT to identify streams which carry AAC audio. The descriptor shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop.

### H.2.1    Syntax and semantics for the AAC descriptor

The `AAC_descriptor` (see table H.1) provides information about individual MPEG-4 AAC, MPEG-4 HE-AAC and MPEG-4 HE-AAC v2 elementary streams to be identified in the PSI PMT sections. The descriptor shall be included in a `program_map_section` at most once in each relevant *ES_info* descriptor loop which describes an elementary stream containing MPEG-4 AAC, MPEG-4 HE-AAC and MPEG-4 HE-AAC v2 audio.

**Table H.1: AAC descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| AAC_descriptor() { | | |
|    descriptor_tag | 8 | uimsbf |
|    descriptor_length | 8 | uimsbf |
|    profile_and_level | 8 | uimsbf |
|    if (descriptor_length > 1) { | | |
|       AAC_type_flag | 1 | bslbf |
|       SAOC_DE_flag | 1 | bslbf |
|       reserved_zero_future_use | 6 | bslbf |
|       if (AAC_type_flag == 0b1 | | |
|          AAC_type | 8 | uimsbf |
|       } | | |
|       for (i=0;i<N;i++) { | | |
|          additional_info_byte | 8 | uimsbf |
|       } | | |
|    } | | |
| } | | |

Semantics for the AAC descriptor:

*profile_and_level*: This 8-bit field specifies the Profile and Level used in MPEG-4 AAC, MPEG-4 HE-AAC or MPEG-4 HE-AAC v2. This field is set to the same value as the MPEG-4_audio_profile_and_level of the MPEG-4_audio_descriptor in ISO/IEC 13818-1 [1].

*AAC_type_flag*: This 1-bit field indicates the presence of the *AAC_type* field. This bit is set to 0b1 if the optional *AAC_type* field is included in the descriptor.

*SAOC_DE_flag*: This 1-bit flag indicates the presence of embedded SAOC-DE parametric data in AAC audio ancillary data and *PES_private_data* according to table H.2.

**Table H.2: SAOC-DE flag coding**

| *SAOC_DE_flag* | Parametric data in AAC audio ancillary data | Parametric data in *PES_private_data* (see note) |
|---|---|---|
| 0b0 | shall not be present | shall not be present |
| 0b1 | SAOC-DE parametric data shall be present | *DE_control_data* may be present |
| NOTE: *PES_private_data* within the PES packet header of the audio component as defined in ETSI TS 101 154 [14], clause E.7.2. | | |

*AAC_type*: This field is set according to table 26 to the value of the *component_type* field when *stream_content* is 0x06. This field shall not be set to 0xA0.

*additional_info_byte*: This is an 8-bit field. The sequence of *additional_info_byte* fields specifies the *additional_info* field. The syntax and semantics of the *additional_info* field may be defined in future versions of the present document.

# Annex I (normative): Assignment and interpretation of the service_type field

## I.1      Background

The *service_type* field is present within both the `service_descriptor` and `service_list_descriptor` and is used to specify the type of a service. The intention of this field is to allow the service provider to describe the nature of the service, e.g. broadcast television, on-demand television, broadcast radio, data broadcast.

This information is deliberately provided at a very high-level within DVB service information to allow the receiver to make some decisions as soon as possible after the discovery of a service (through re-scan or some other mechanism) about how, and indeed whether, to present the service to the viewer for selection.

For example, information provided by the *service_type* field about the nature of a service can be used to group services into dedicated service lists for presentation to the viewer, e.g. separate television and radio lists.

The *service_type* field is not meant to override information provided at lower levels within SI or within PSI, such as the assignment of *stream_type* for a component within the PMT or the actual coding within the component itself, particularly with respect to the decoding and presentation of components of a service. It is provided because such low-level information may be difficult to interpret, slow to acquire, or (in the case of time-exclusive services) not even present at the time of service discovery.

> NOTE:       The receiver may also consider other factors, such as conditional access, when making such decisions. However, this is beyond the scope of the present document.

Broadcasters are recommended to use newly introduced service types when services are not intended to be listed on older receivers, and to use existing service types when services are intended to be listed on older receivers.

## I.2      Assignment of service_type

### I.2.0      General principles

For some services the assignment of a *service_type* from table 89 may be obvious, e.g. MPEG-2 HD digital television service. However, the decision is not always so straightforward.

Since the SDT is intended to be used by receivers to build service lists, the *service_type* shall be changed only when required due to operational changes (for example a reconfiguration of the corresponding multiplex or delivery system), to avoid unintended service list updates.

### I.2.1      service_type "digital television service" (0x01)

In the generic case this *service_type* provides no explicit indication to the receiver about the way in which the components of a service have been encoded.

> NOTE 1:     Of course, in the case of a specific platform a particular encoding could be implicitly linked to this *service_type* and so inferred by the receiver. However, any such arrangement is beyond the scope of the present document.

As indicated by note 1 in table 89, this *service_type* should be used for MPEG-2 SD digital television service. However, it may also be used for services using other encodings, including encodings that have a specific entry elsewhere in table 89, e.g. MPEG-2 HD digital television service.

> NOTE 2:     DVB has deliberately not refined the definition of this *service_type* from "digital television service" to "MPEG-2 SD digital television service" due pre-existing use in the context of other (non-MPEG-2 SD) encodings.

On the assumption that all receivers will be able to decode and present MPEG-2 SD encoded material, all receivers will present any service assigned this *service_type* to the viewer for selection on the basis that it may be MPEG-2 SD coded material. However, as described above, this may not be the case and the receiver may not support the actual encoding used. This inability for the receiver to determine whether or not it will actually be able to decode and present a service assigned this *service_type* means that the service provider needs to allocate it with care depending on the viewer experience it wishes to achieve.

EXAMPLE: Consider a platform where some services are based on MPEG-2 SD encoding and others are based on MPEG-2 HD encoding, both of which are delivered to a mixed population of MPEG-2 SD-only and MPEG-2 SD/HD receivers.

For a service based on MPEG-2 SD encoding the assignment of *service_type* is obvious: 0x01 ("digital television service").

However, for a service based on MPEG-2 HD encoding the assignment of *service_type* depends on whether the service provider wants the service to be included in any service list presented to viewers of MPEG-2 SD-only receivers, even though they will not actually be able to view the service if selected. If this is the desired viewer experience then the service should be allocated *service_type* 0x01 ("digital television service"). If, however, the desired viewer experience is only to list services that the viewer of an MPEG-2 SD-only receiver is actually able to view then the service should be allocated *service_type* 0x11 ("MPEG-2 HD digital television service").

This *service_type* may also be allocated to a service that contains both an MPEG-2 SD encoding and an alternative encoding (e.g. an MPEG-4 HD) of the same material. This is reasonable on the assumption that all receivers will be able to decode and present MPEG-2 SD encoded material, hence the viewer will at least be presented with the MPEG-2 SD coded form. However, depending on the capabilities of the receiver in use the viewer may be presented with the alternative, typically superior, coded form.

NOTE 3: The components used for the different encodings can be discriminated between at the point of decode by the assigned values for *stream_type* in PSI and/or use of the `component_descriptor` in SI.

## I.2.2 service_type "H.264/AVC" (various)

In previous versions of the present document the Advanced Video Coding for Generic Audiovisual Services (H.264/AVC) *service_type* values were called "advanced codec" and were allocated so as to be able to indicate that the video component of a service had been encoded using something other than MPEG-2. However, many IRD implementations had assumed such services would always contain an H.264/AVC video component. To avoid confusion the *service_type* values have been renamed to indicate H.264/AVC only.

The H.264/AVC *service_type* values shall hence only be used for services that provide an H.264/AVC video component. They may also be used with newer codecs, provided the service includes a video component which is backwards compatible with an H.264/AVC IRD.

## I.2.3 service_type "H.264/AVC frame compatible stereoscopic HD" (various)

The H.264/AVC frame compatible stereoscopic HD values allow a broadcaster to signal that a service operates (primarily) as a frame compatible stereoscopic service. The use of these values requires careful consideration of the consequences for legacy receiver populations, which as a result may ignore these services. Therefore, a broadcaster may instead choose to signal a frame compatible stereoscopic service as a normal HD service, and use alternative signalling to indicate that the service (or event) is in a frame compatible stereoscopic format.

Further discussion of the signalling of frame compatible stereoscopic services is provided in ETSI TS 101 547-2 [16].

# I.2.4 service_type "advanced codec digital radio sound service" (0x0A)

The advanced codec digital radio sound *service_type* has been allocated to indicate that an audio service has been encoded using something other than MPEG-1 Layer 2. The assignment of this *service_type* provides a generic indication of the use of some advanced audio codec without stating specifically which one. As such, on its own, it does not fully allow a receiver to determine that it is able to decode and present a service assigned this *service_type*.

NOTE:     In the case of a specific platform a particular encoding may be implicitly linked to this *service_type* and so inferred by the receiver. However, any such arrangement is beyond the scope of the present document.

Where a service is assigned this *service_type*, the `component_descriptor` should be used to indicate the particular advanced codec used. This allows a receiver to unambiguously determine whether or not it will be able to decode and present the service and handle as appropriate.

# I.2.5 service_type "HEVC digital television service" (0x1F)

## I.2.5.0 General principles

The HEVC digital television *service_type* has been allocated, and shall only be used for all the possible types of an HEVC video services that are decodable by HEVC High Definition TeleVision (HDTV) IRD and HEVC UHDTV IRD as defined in ETSI TS 101 154 [14]. The *service_type* alone does not fully allow a receiver to determine that it is able to decode a service assigned the 0x1F *service_type*. Where a service is assigned this *service_type*, the `component_descriptor` shall hence be used to indicate the specific type of the HEVC service. This allows a receiver to unambiguously determine whether or not it will be able to decode and present the service and handle as appropriate.

One or more `component_descriptor` shall be present in the SDT entry of each HEVC digital television service. There shall be one and only one `component_descriptor` using values listed in table I.1, which is derived from table 26. Other component descriptors may also be present in the SDT, but no `component_descriptor` using values from table I.9 shall be present.

**Table I.1: HEVC component types from table 26 for service_type 0x1F**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0x9 | 0x0 | 0x00 | HEVC Main Profile high definition video, 50 Hz |
| | | 0x01 | HEVC Main 10 Profile high definition video, 50 Hz |
| | | 0x02 | HEVC Main Profile high definition video, 60 Hz |
| | | 0x03 | HEVC Main 10 Profile high definition video, 60 Hz |
| | | 0x04 | HEVC ultra high definition video (see note) conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]: <br> • "SDR frame rate up to 60 Hz"; <br> • "SDR HFR dual PID with temporal scalability"; <br> • "HDR with HLG10 frame rate up to 60 Hz"; <br> • "HDR with HLG10 HFR dual PID and temporal scalability". |
| NOTE: This value should be used for HLG10 HDR services, and/or HFR services with dual PID and temporal scalability as defined in ETSI TS 101 154 [14]. See also clause I.2.5.2. | | | |

Since `component_descriptor` when used within the SDT can be used by receivers to build service lists, it is recommended that they only be modified occasionally. If broadcasters wish to indicate a different HEVC bit stream conformance point in use for a particular event, but which is also decodable by the receiver, it is recommended they signal this via one or more informative `component_descriptor` in the EIT. These `component_descriptor` should indicate the highest HEVC receiver profile that will be used on the service. Such a service may however also contain programmes encoded at a lower profile or resolution. Broadcasters should note that by signalling a particular profile, they may prevent receivers implementing a lower profile from acquiring that service. However, a broadcaster signalling a lower profile than the maximum profile actually used by the service, will result in programmes encoded with such a higher profile not being viewable on lower profile receivers.

The *service_type* value shall not be changed during a running service, i.e. a service with *service_type* 0x1F shall not be changed to *service_type* 0x20, to signal the carriage of content that contains any component with HDR or a frame rate above 60 Hz.

## I.2.5.1 Signalling for service frame compatible plano-stereoscopic 3DTV for HEVC coded services

Table I.4 shows the usage of the attributes specified in previous clauses for the Service Frame Compatible (SFC)-3DTV service classes listed in clause 4.1 of ETSI TS 101 547-4 [18]. The table is not intended to represent an exhaustive list of possibilities nor specify the only signalling option for each form of 3DTV service. Rather, it is meant as guidance and for raising awareness of possible issues that need to be considered with the deployment of SFC-3DTV services.

The present document describes a single *service_type* for HEVC, and uses component_descriptor to provide detailed differentiation between the service content. Thus, to signal SFC-3DTV services that are intended to be listed by legacy Two Dimensional (2D) receivers (i.e. backwards compatible signalling), a combination of a *service_type* indicating HEVC, and an SDT component_descriptor indicating one of the four high definition HEVC *stream_content*/*stream_content_ext*/*component_type* values as listed in table I.2 is used. Other component_descriptor may also be present in the SDT, but no component_descriptor using values from table I.5 or table I.9 shall be present.

**Table I.2: HEVC component types from table 26 for SFC-3DTV**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0x9 | 0x0 | 0x00 | HEVC Main Profile high definition video, 50 Hz |
| | | 0x01 | HEVC Main 10 Profile high definition video, 50 Hz |
| | | 0x02 | HEVC Main Profile high definition video, 60 Hz |
| | | 0x03 | HEVC Main 10 Profile high definition video, 60 Hz |

To signal SFC-3DTV events that are intended to be listed by legacy 2D receivers (i.e. backwards compatible signalling), an additional component_descriptor in the EIT is required to indicate the SFC-3DTV is plano-stereoscopic to SFC-3DTV cognisant receivers. This additional component_descriptor shall be placed in the EIT. Only one additional component_descriptor is currently defined for HEVC SFC-3DTV services and shown in table I.3. Other values maybe defined later.

**Table I.3: Currently defined HEVC plano-stereoscopic frame-packing component types from table 26**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0xB | 0xF | 0x03 | plano-stereoscopic top and bottom (TaB) frame-packing |

NOTE: Information about the current coding parameters of SFC-3DTV video is provided at the elementary stream level as specified in ETSI TS 101 547-4 [18] and ETSI TS 101 154 [14].

IRDs already existing in the field might not recognize the newly defined SFC-3DTV signalling, and this legacy may be supported through the video level signalling as described in clause 5.14.1.5.6 of ETSI TS 101 154 [14], in which case a 2D picture may be extracted from the 3DTV content by receivers supporting the HEVC default display window.

**Table I.4: Summary of service compatible frame compatible plano-stereoscopic
3DTV service signalling scenarios**

| Service category | SDT | | EIT `component_descriptor` | Description |
|---|---|---|---|---|
| | `service_type` | `component_descriptor` | | |
| occasional 3DTV service | 0x1F (HEVC) | One from table I.2 (see note) | For 3DTV events one SFC-3DTV related from table I.3 (see note); other `component_descriptor` may be present, including multiple for the same elementary stream. | This is the recommended signalling; 3DTV events are signalled via EIT |
| NOTE: | SFC-3DTV signalling requires two `component_descriptor`, one in the SDT, and one in the EIT that might be modified on an event basis. The `component_descriptor` in the SDT indicates the type of HEVC video using *stream_content* set to 0x9, *stream_content_ext* set to 0x0, and the *component_type* set to a value in the range 0x00 to 0x03. The plano-stereoscopic top and bottom (TaB) frame-packing format is indicated by an additional `component_descriptor` in the EIT with *stream_content* 0xB, *stream_content_ext* 0xF, and *component_type* 0x03. | | | |

## I.2.5.2 Signalling for HDR and/or frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz, but with a HEVC half frame rate temporal video sub-bitstream frame rate lower than or equal to 60 Hz

An Ultra high definition video service with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz containing a half frame rate HEVC temporal video sub-bitstream and/or HLG10 HDR is decodable by HEVC_UHDTV_IRD as defined in ETSI TS 101 154 [14]. A `component_descriptor` shall be present in the SDT entry of such a service using the values defined in table I.5. Other `component_descriptor` may also be present in the SDT, but no `component_descriptor` using values from table I.2 or table I.9 shall be present.

**Table I.5: HEVC component types from table 26
for ultra high definition video using service_type 0x1F**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0x9 | 0x0 | 0x04 | HEVC ultra high definition video conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]: <br>• "SDR frame rate up to 60 Hz"; <br>• "SDR HFR dual PID with temporal scalability"; <br>• "HDR with HLG10 frame rate up to 60 Hz"; <br>• "HDR with HLG10 HFR dual PID and temporal scalability". |

Additional `component_descriptor` may be present in the SDT and/or in the EIT to allow receiver to distinguish between services or events containing a HEVC UHDTV bitstream (as defined in ETSI TS 101 154 [14]) and those which exceed those requirements but are still decodable by ETSI TS 101 154 [14]. These `component_descriptor` are shown in table I.6. HEVC ultra high definition IRDs that are neither HDR nor capable of decoding a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz can ignore the `component_descriptor` from table I.6 and decode the service as an SDR with a frame rate of up to 60 Hz.

**Table I.6: HEVC component types from table 26 for a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz,
or 120 Hz and HDR using service_type 0x1F**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0xB | 0xF | 0x04 | HLG10HDR |
| | | 0x05 | HEVC temporal video subset for a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz |

## I.2.5.3　Spatial, temporal, and dynamic range characteristics

Additional component_descriptor may also be present in the SDT and EIT to inform on the source spatial, temporal and dynamic range characteristics. These component_descriptor (as shown in table I.7) are intended to be used in the EIT when the broadcaster wants to indicate the video characteristics for a particular event are less than the capabilities of the service indicated by the *service_type* and component_descriptor in the SDT.

**Table I.7: Component types from table 26 to inform on the source's spatial, temporal, and dynamic range characteristics**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0x3 | n/a | 0x40 | Video spatial resolution has been upscaled from lower resolution source material |
| | | 0x41 | Video is SDR |
| | | 0x42 | Video is HDR remapped from SDR source material |
| | | 0x43 | Video is HDR up-converted from SDR source material |
| | | 0x44 | Video is standard frame rate, less than or equal to 60 Hz |
| | | 0x45 | High frame rate video generated from lower frame rate source material |

## I.2.5.4　Summary of signalling for different bitstream profiles using service_type 0x1F

**Table I.8: Signalling for HEVC bitstream profiles**

| HEVC profile | service_type | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | stream_content | stream_content_ext | component_type | stream_content | stream_content_ext | component_type |
| HEVC Main Profile high definition video, 50 Hz | 0x1F | 0x9 | 0x0 | 0x00 | | | |
| HEVC Main 10 Profile high definition video, 50 Hz | 0x1F | 0x9 | 0x0 | 0x01 | | | |
| HEVC Main Profile high definition video, 60 Hz | 0x1F | 0x9 | 0x0 | 0x02 | | | |
| HEVC Main 10 Profile high definition video, 60 Hz | 0x1F | 0x9 | 0x0 | 0x03 | | | |
| HEVC ultra high definition video conformant to the conformance point "SDR frame rate up to 60 Hz" as is defined in table 18b in ETSI TS 101 154 [14] | 0x1F | 0x9 | 0x0 | 0x04 | | | |
| HEVC ultra high definition video conformant to the conformance point "SDR HFR dual PID and temporal scalability" as is defined in table 18b in ETSI TS 101 154 [14] | 0x1F | 0x9 | 0x0 | 0x04 | 0xB | 0xF | 0x05 |
| HEVC ultra high definition video conformant to the conformance point "HDR with HLG10 frame rate up to 60 Hz" as is defined in table 18b in ETSI TS 101 154 [14] | 0x1F | 0x9 | 0x0 | 0x04 | 0xB | 0xF | 0x04 |
| HEVC ultra high definition video conformant to the conformance point "HDR with HLG10 HFR dual PID and temporal scalability" as is defined in table 18b in ETSI TS 101 154 [14] (see note 2) | 0x1F | 0x9 | 0x0 | 0x04 | 0xB / 0xB | 0xF / 0xF | 0x04 / 0x05 |

| HEVC profile | *service_type* | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| NOTE 1: These component_descriptor are optional to broadcast. A broadcaster can signal them if they wish for receivers to be able to group services by their HEVC bit stream profile. There are no requirements for receivers to use these. | | | | | | | |
| NOTE 2: Both optional component descriptors are intended to be present. | | | | | | | |

# I.2.6        service_type "HEVC UHD digital television service" (0x20)

## I.2.6.1    General principles

In order to avoid compatibility issues with already deployed HEVC_HDTV_IRD and HEVC_UHDTV_IRD, a new *service_type* is defined for the following HEVC digital television services, which cannot be decoded by HEVC_UHDTV_IRD as defined in ETSI TS 101 154 [14]:

- with a resolution up to 3 840 x 2 160, HDR and/or a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz; or

- with a resolution greater than 3 840 x 2 160, SDR or HDR, with a frame rate up to 60 Hz.

The *service_type* 0x20 shall only be used for all the possible types of an HEVC video service that are not decodable by a HEVC_UHDTV_IRD (as defined in ETSI TS 101 154 [14]). The *service_type* on its own does not fully allow a receiver to determine that it is able to decode a service assigned this *service_type*. Where a service is assigned this *service_type*, the component_descriptor shall hence be used to indicate the specific type of the HEVC digital television service. This allows a receiver to unambiguously determine whether or not it will be able to decode and present the service.

One or more component_descriptor shall be present in the SDT entry of each HEVC digital television service with *service_type* 0x20. There shall be one and only one component_descriptor using values from table I.9, a subset derived from table 26. Other component descriptors may also be present in the SDT (such as those defined in table I.7), but no component_descriptor using values from table I.1 shall be present.

**Table I.9: HEVC component types from table 26 for service_type 0x20**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0x9 | 0x0 | 0x05 | HEVC ultra high definition video with PQ10 HDR with a frame rate lower than or equal to 60 Hz conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "HDR with PQ10 frame rate up to 60 Hz resolution up to 3 840 x 2 160"<br>or<br>HEVC ultra high definition video with PQ10 HDR with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz with a half frame rate HEVC temporal video sub-bit-stream conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "HDR with PQ10 HFR dual PID and temporal scalability resolution up to 3 840 x 2 160". |
| | | 0x06 | HEVC ultra high definition video, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "SDR HFR single PID resolution up to 3 840 x 2 160";<br>• "HDR with HLG10 HFR single PID resolution up to 3 840 x 2 160". |
| | | 0x07 | HEVC ultra high definition video with PQ10 HDR, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to one of the following UHDTV bit stream conformance points defined in table 18b in ETSI TS 101 154 [14]:<br>• "PQ10 HFR single PID resolution up to 3 840 x 2 160" |
| | | 0x08 | HEVC ultra high definition video with a resolution up to 7 680 x 4 320 conformant to one of the following UHDTV2 bit stream conformance point defined in table 18b in ETSI TS 101 154 [14]:<br>• "SDR frame rate up to 60 Hz resolution up to 7 680 x 4 320";<br>• "HDR with PQ10 frame rate up to 60 Hz resolution up to 7 680 x 4 320";<br>• "HDR with HLG10 frame rate up to 60 Hz resolution up to 7 680 x 4 320". |

Since `component_descriptor` when used within the SDT can be used by receivers to build service lists, it is recommended that they only be modified occasionally. If broadcasters wish to indicate a different HEVC bit stream conformance point in use for a particular event, but which is also decodable by the receiver, it is recommended they signal this via informative `component_descriptor` in the EIT.

Additional `component_descriptor` may be present in the SDT and/or in the EIT to allow receiver to distinguish between services or events containing a half frame rate HEVC temporal video sub-bitstream or HLG10 HDR. These `component_descriptor` are listed in table I.10.

Additional `component_descriptor` may also be present in the SDT and EIT to inform on the source spatial, temporal and dynamic range characteristics, as described in clause I.2.5.3.

For HEVC ultra high definition video with PQ10 HDR bitstreams additional `component_descriptor` may also be present in the SDT and EIT to indicate the presence of SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14]. These `component_descriptor` are listed in table I.11. An HEVC ultra high definition video with PQ10 HDR bitstream may simultaneously convey DMI in multiple formats, this should be indicated by the presence of multiple `component_descriptor` from table I.11.

The *service_type* value shall not be changed during a running service, even in the case when a service with *service_type* 0x20 intermittently carries content that does not contain any component with HDR, nor a frame rate above 60 Hz, whereas such content would be compatible with a service with *service_type* 0x1F.

**Table I.10: HEVC component types from table 26 indicating the presence of a half frame rate HEVC temporal video sub-bitstream or HLG10 HDR using service_type 0x20**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0xB | 0xF | 0x04 | HLG10 HDR |
| | | 0x05 | HEVC temporal video subset for a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz or 120 Hz |

**Table I.11: HEVC component types from table 26 indicating the presence of DMI metadata**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0xB | 0xF | 0x06 | SMPTE ST 2094-10 DMI format as defined in clause 5.14.4.4.3.4.3 of ETSI TS 101 154 [14] |
| | | 0x07 | SL-HDR2 DMI format as defined in clause 5.14.4.4.3.4.4 of ETSI TS 101 154 [14] |
| | | 0x08 | SMPTE ST 2094-40 DMI format as defined in clause 5.14.4.4.3.4.5 of ETSI TS 101 154 [14] |

## I.2.6.2    Summary of signalling for different bitstream profiles using service_type 0x20

**Table I.12: Signalling for HEVC bitstream profiles**

| HEVC profile | *service_type* | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| HEVC ultra high definition video, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "SDR HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x06 | | | |
| HEVC ultra high definition video, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with HLG10 HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x06 | 0xB | 0xF | 0x04 |
| HEVC ultra high definition video with PQ10 HDR with a frame rate lower than or equal to 60 Hz conformant to the conformance point "HDR with PQ10 frame rate up to 60 Hz resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x05 | | | |

| HEVC profile | service_type | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| | | stream_content | stream_content_ext | component_type | stream_content | stream_content_ext | component_type |
| HEVC ultra high definition video with PQ10 HDR with a frame rate lower than or equal to 60 Hz conformant to the conformance point "HDR with PQ10 frame rate up to 60 Hz resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2) | 0x20 | 0x9 | 0x0 | 0x05 | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |
| HEVC ultra high definition video with PQ10 HDR with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz containing a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR dual PID and temporal scalability resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x05 | 0xB | 0xF | 0x05 |
| HEVC ultra high definition video with PQ10 HDR with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz containing a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR dual PID and temporal scalability resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2) | 0x20 | 0x9 | 0x0 | 0x05 | 0xB | 0xF | 0x05 |
| | | | | | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |
| HEVC ultra high definition video with PQ10 HDR, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x07 | | | |
| HEVC ultra high definition video with PQ10 HDR, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2) | 0x20 | 0x9 | 0x0 | 0x7 | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz conformant to the conformance point "SDR Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] (see note 3) | 0x20 | 0x9 | 0x0 | 0x08 | | | |

| HEVC profile | service_type | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz conformant to the conformance point "HDR with PQ10 Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] (see note 3) | 0x20 | 0x9 | 0x0 | 0x08 | | | |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz, with PQ10 HDR conformant to the conformance point "HDR with PQ10 Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2 and note 3) | 0x20 | 0x9 | 0x0 | 0x08 | 0xB 0xB 0xB | 0xF 0xF 0xF | 0x06 0x07 0x08 |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz, with HLG10 HDR conformant to the conformance point "HDR with HLG10 Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] (see note 3) | 0x20 | 0x9 | 0x0 | 0x08 | 0xB | 0xF | 0x04 |

NOTE 1: These component_descriptor are optional to broadcast. A broadcaster can signal them if they wish for receivers to be able to group services by their HEVC bit stream profile. There are no requirements for receivers to use these.

NOTE 2: The component_descriptor (from table I.11) indicating the presence of DMI metadata formats should only be signalled for the formats that are likely to be present.

NOTE 3: These bitstream conformance points intentionally use the same signalling as they are intended to be decoded by the same IRD conformance point.

## I.2.7    Summary of signalling for HEVC bitstream profiles using service_type 0x1F or 0x20

Table  I.13 summarizes the different ultra high definition bitstream signalling for convenience.

**Table I.13: Signalling for HEVC bitstream profiles**

| HEVC profile | *service_type* | **Mandatory SDT** component_descriptor | | | **Optional SDT** component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| HEVC Main Profile high definition video, 50 Hz | 0x1F | 0x9 | 0x0 | 0x00 | | | |
| HEVC Main 10 Profile high definition video, 50 Hz | 0x1F | 0x9 | 0x0 | 0x01 | | | |
| HEVC Main Profile high definition video, 60 Hz | 0x1F | 0x9 | 0x0 | 0x02 | | | |
| HEVC Main 10 Profile high definition video, 60 Hz | 0x1F | 0x9 | 0x0 | 0x03 | | | |
| HEVC ultra high definition video conformant to the conformance point "SDR frame rate up to 60 Hz" as is defined in table 18b in ETSI TS 101 154 [14] | 0x1F | 0x9 | 0x0 | 0x04 | | | |
| HEVC ultra high definition video conformant to the conformance point "SDR HFR dual PID and temporal scalability" as is defined in table 18b in ETSI TS 101 154 [14] | 0x1F | 0x9 | 0x0 | 0x04 | 0xB | 0xF | 0x05 |
| HEVC ultra high definition video conformant to the conformance point "HDR with HLG10 frame rate up to 60 Hz" as is defined in table 18b in ETSI TS 101 154 [14] | 0x1F | 0x9 | 0x0 | 0x04 | 0xB | 0xF | 0x04 |
| HEVC ultra high definition video conformant to the conformance point "HDR with HLG10 HFR dual PID and temporal scalability" as is defined in table 18b in ETSI TS 101 154 [14] (see note 4) | 0x1F | 0x9 | 0x0 | 0x04 | 0xB<br>0xB | 0xF<br>0xF | 0x04<br>0x05 |
| HEVC ultra high definition video, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "SDRHFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x06 | | | |
| HEVC ultra high definition video, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with HLG10 HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x06 | 0xB | 0xF | 0x04 |
| HEVC ultra high definition video with PQ10 HDR with a frame rate lower than or equal to 60 Hz conformant to the conformance point "HDR with PQ10 frame rate up to 60 Hz resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x05 | | | |

| HEVC profile | *service_type* | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| HEVC ultra high definition video with PQ10HDR with a frame rate lower than or equal to 60 Hz conformant to the conformance point "HDR with PQ10 frame rate up to 60 Hz resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2) | 0x20 | 0x9 | 0x0 | 0x05 | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |
| HEVC ultra high definition video with PQ10 HDR with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz containing a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR dual PID and temporal scalability resolution up to 3 840 x 2160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x05 | 0xB | 0xF | 0x05 |
| HEVC ultra high definition video with PQ10 HDR with a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz containing a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR dual PID and temporal scalability resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2) | 0x20 | 0x9 | 0x0 | 0x05 | 0xB | 0xF | 0x05 |
| | | | | | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |
| HEVC ultra high definition video with PQ10 HDR, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] | 0x20 | 0x9 | 0x0 | 0x07 | | | |
| HEVC ultra high definition video with PQ10 HDR, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz without a half frame rate HEVC temporal video sub-bit-stream conformant to the conformance point "HDR with PQ10 HFR single PID resolution up to 3 840 x 2 160" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2) | 0x20 | 0x9 | 0x0 | 0x7 | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz conformant to the conformance point "SDR Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] (see note 3) | 0x20 | 0x9 | 0x0 | 0x08 | | | |

| HEVC profile | *service_type* | Mandatory SDT component_descriptor | | | Optional SDT component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz conformant to the conformance point "HDR with PQ10 Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] (see note 3) | 0x20 | 0x9 | 0x0 | 0x08 | | | |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60 Hz, with PQ10 HDR conformant to the conformance point "HDR with PQ10 Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] and containing SEI messages carrying DMI as specified in clause 5.14.4.4.3.4 of ETSI TS 101 154 [14] (see note 2 and note 3) | 0x20 | 0x9 | 0x0 | 0x08 | 0xB<br>0xB<br>0xB | 0xF<br>0xF<br>0xF | 0x06<br>0x07<br>0x08 |
| HEVC ultra high definition video with a resolution up to 7 680 x 4 320, frame rate up to 60Hz, with HLG10 HDR conformant to the conformance point "HDR with HLG10 Frame Rate up to 60 Hz Resolution up to 7 680 x 4 320" as is defined in table 18b in ETSI TS 101 154 [14] (see note 3) | 0x20 | 0x9 | 0x0 | 0x08 | 0xB | 0xF | 0x04 |

NOTE 1: These component_descriptor are optional to broadcast. A broadcaster can signal them if they wish for receivers to be able to group services by their HEVC bit stream profile. There are no requirements for receivers to use these.

NOTE 2: The component_descriptor (from table I.11) indicating the presence of DMI metadata formats should only be signalled for the formats that are likely to be present.

NOTE 3: These bitstream conformance points intentionally use the same signalling as they are intended to be decoded by the same IRD conformance point.

NOTE 4: Both optional component descriptors are intended to be present.

## I.2.8     service_type "VVC digital television service" (0x21)

### I.2.8.1    General principles

The VVC digital television *service_type* has been allocated, and shall only be used for all the possible types of VVC video services that are decodable by VVC IRD conformance points as defined in ETSI TS 101 154 [14]. The *service_type* alone does not fully allow a receiver to determine that it is able to decode a service assigned the 0x21 *service_type*. Where a service is assigned this *service_type*, the component_descriptor in the SDT shall hence be used to indicate the specific type of the VVC service. This allows a receiver to unambiguously determine whether or not it will be able to decode and present the service and handle as appropriate.

One or more component_descriptor shall be present in the SDT entry of each VVC digital television service. There shall be one and only one component_descriptor using values listed in table I.1, which is derived from table 26. Other component descriptors may also be present in the SDT.

**Table I.14: VVC component types from table 26 for service_type 0x21**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0x9 | 0x0 | 0x10 | VVC Main 10 Profile with resolution up to 3 840 x 2 160 frame rate up to 60 Hz conformant to the VVC HDR UHDTV-1 bitstream conformance point defined in clause 5.15.2 in ETSI TS 101 154 [14]. |
| | | 0x11 | VVC Main 10 Profile with resolution up to 3 840 x 2 160 High Frame Rate of 100 Hz or 120 Hz conformant to the VVC HDR HFR UHDTV-1 bitstream conformance point defined in clause 5.15.3 in ETSI TS 101 154 [14]. |
| | | 0x12 | VVC Main 10 Profile with resolution up to 7 680 x 4 320 frame rate up to 60 Hz conformant to the VVC HDR UHDTV-2 bitstream conformance point defined in clause 5.15.4 in ETSI TS 101 154 [14]. |
| | | 0x13 | VVC Main 10 Profile with resolution up to 7 680 x 4 320 High Frame Rate of 100 Hz or 120 Hz conformant to the VVC HDR HFR UHDTV-2 bitstream conformance point defined in clause 5.15.5 in ETSI TS 101 154 [14]. |

Since component_descriptor when used within the SDT are typically used by receivers to build service lists, it is recommended that they only be modified occasionally. The component_descriptor within the SDT should indicate the highest VVC receiver profile that will be used on the service. Such a service may however also contain programmes encoded at a lower profile or resolution. Broadcasters should note that by signalling a particular profile, they may prevent receivers implementing a lower profile from acquiring that service. However, if a broadcaster signals a lower profile than the maximum profile actually used by the service, it will result in programmes encoded with such a higher profile not being viewable on lower profile receivers.

The *service_type* value shall not be changed during a running service.

## I.2.8.2    Spatial, temporal and dynamic range characteristics

Additional component_descriptor may also be present in the SDT and EIT to inform on the source spatial, temporal and dynamic range characteristics. These component_descriptor (as shown in table I.15) are intended to be used in the SDT when a broadcaster wants to indicate more information about the spatial and dynamic range characteristics of the whole service. These component_descriptor (as shown in table I.15) are intended to be used in the EIT when the broadcaster wants to indicate the video characteristics for a particular event are less than the capabilities of the service indicated by the *service_type* and component_descriptor in the SDT.

**Table I.15: Component types from table 26 to inform on the source's spatial, temporal, and dynamic range characteristics**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0x3 | n/a | 0x40 | Video spatial resolution has been upscaled from lower resolution source material |
| | | 0x41 | Video is SDR |
| | | 0x42 | Video is HDR remapped from SDR source material |
| | | 0x43 | Video is HDR up-converted from SDR source material |
| | | 0x44 | Video is standard frame rate, less than or equal to 60 Hz |
| | | 0x45 | High frame rate video generated from lower frame rate source material |
| 0xB | 0xF | 0x04 | HLG10 HDR |
| | | 0x06 | SMPTE ST 2094-10 DMI format as defined ETSI TS 101 154 [14] |
| | | 0x07 | SL-HDR2 DMI format as defined in ETSI TS 101 154 [14] |
| | | 0x08 | SMPTE ST 2094-40 DMI format as defined in ETSI TS 101 154 [14] |
| | | 0x09 | PQ10 HDR |

## I.2.8.3　Examples of signalling for different bitstream profiles using service_type 0x21

**Table I.16: Signalling for VVC bitstream profiles**

| VVC profile | service_type | Mandatory SDT component_descriptor | | | Optional component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | stream_content | stream_content_ext | component_type | stream_content | stream_content_ext | component_type |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) standard dynamic range frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x10 | 0x3 | n/a | 0x41 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range HLG frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x10 | 0xB | 0xF | 0x04 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x10 | 0xB | 0xF | 0x09 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-1" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI. | 0x21 | 0x9 | 0x0 | 0x10 | 0xB / 0xB / 0xB | 0xF / 0xF / 0xF | 0x06 / 0x07 / 0x08 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) standard dynamic range frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x11 | 0x3 | n/a | 0x41 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range HLG frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x11 | 0xB | 0xF | 0x04 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x11 | 0xB | 0xF | 0x09 |
| VVC bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI. | 0x21 | 0x9 | 0x0 | 0x11 | 0xB / 0xB / 0xB | 0xF / 0xF / 0xF | 0x06 / 0x07 / 0x08 |

| VVC profile | *service_type* | Mandatory SDT component_descriptor | | | Optional component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| VVC bitstream with resolution up to 7 680 x 4 320 standard dynamic range frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x12 | 0x3 | n/a | 0x41 |
| VVC bitstream with resolution up to 7 680 x 4 320 high dynamic range HLG frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x12 | 0xB | 0xF | 0x04 |
| VVC bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x12 | 0xB | 0xF | 0x09 |
| VVC bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "VVC HDR UHDTV-2" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI | 0x21 | 0x9 | 0x0 | 0x12 | 0xB<br>0xB<br>0xB | 0xF<br>0xF<br>0xF | 0x06<br>0x07<br>0x08 |
| VVC bitstream with resolution up to 7 680 x 4 320 standard dynamic range frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x13 | 0x3 | n/a | 0x41 |
| VVC bitstream with resolution up to 7 680 x 4 320 high dynamic range HLG frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x13 | 0xB | 0xF | 0x04 |
| VVC bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 120Hz conformant to the conformance point "VVC HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x21 | 0x9 | 0x0 | 0x13 | 0xB | 0xF | 0x09 |
| VVC bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "VVC HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI | 0x21 | 0x9 | 0x0 | 0x13 | 0xB<br>0xB<br>0xB | 0xF<br>0xF<br>0xF | 0x06<br>0x07<br>0x08 |
| NOTE: These component_descriptor are optional to broadcast. A broadcaster can signal them if they wish for receivers to be able to group services by their VVC bit stream profile. There are no requirements for receivers to use these. | | | | | | | |

# I.2.9     service_type "AVS3 digital television service" (0x22)

## I.2.9.1    General principles

The AVS3 digital television *service_type* has been allocated, and shall only be used for all the possible types of AVS3 video services that are decodable by AVS3 IRD conformance points as defined in ETSI TS 101 154 [14]. The *service_type* alone does not fully allow a receiver to determine that it is able to decode a service assigned the 0x22 *service_type*. Where a service is assigned this *service_type*, the component_descriptor in the SDT shall hence be used to indicate the specific type of the AVS3 service. This allows a receiver to unambiguously determine whether or not it will be able to decode and present the service and handle as appropriate.

One or more component_descriptor shall be present in the SDT entry of each AVS3 digital television service. There shall be one and only one component_descriptor using values listed in table I.1, which is derived from table 26. Other component descriptors may also be present in the SDT.

**Table I.17: AVS3 component types from table 26 for service_type 0x22**

| stream_content | stream_content_ext | component_type | Description |
|---|---|---|---|
| 0x9 | 0x0 | 0x20 | AVS3 High 10 Profile with resolution up to 3 840 x 2 160 frame rate up to 60 Hz conformant to the AVS3 HDR UHDTV-1 bitstream conformance point defined in clause 5.16.3 in ETSI TS 101 154 [14] |
| | | 0x21 | AVS3 High 10 Profile with resolution up to 3 840 x 2 160 High Frame Rate of 100 Hz or 120Hz conformant to the AVS3 HDR HFR UHDTV-1 bitstream conformance point defined in clause 5.16.4 in ETSI TS 101 154 [14] |
| | | 0x22 | AVS3 High 10 Profile with resolution up to 7 680 x 4 320 frame rate up to 60 Hz conformant to the AVS3 HDR UHDTV-2 bitstream conformance point defined in clause 5.16.5 in ETSI TS 101 154 [14] |
| | | 0x23 | AVS3 High 10 Profile with resolution up to 7 680 x 4 320 High Frame Rate of 100 Hz or 120 Hz conformant to the AVS3 HDR HFR UHDTV-2 bitstream conformance point defined in clause 5.16.6 in ETSI TS 101 154 [14] |

Since component_descriptor when used within the SDT are typically used by receivers to build service lists, it is recommended that they only be modified occasionally. The component_descriptor within the SDT should indicate the highest AVS3 receiver profile that will be used on the service. Such a service may however also contain programmes encoded at a lower profile or resolution. Broadcasters should note that by signalling a particular profile, they may prevent receivers implementing a lower profile from acquiring that service. However, if a broadcaster signals a lower profile than the maximum profile actually used by the service, it will result in programmes encoded with such a higher profile not being viewable on lower profile receivers.

The *service_type* value shall not be changed during a running service.

## I.2.9.2    Spatial, temporal and dynamic range characteristics

Additional component_descriptor may also be present in the SDT and EIT to inform on the source spatial, temporal and dynamic range characteristics. These component_descriptor (as shown in table I.1) are intended to be used in the SDT when a broadcaster wants to indicate more information about the spatial and dynamic range characteristics of the whole service. These component_descriptor (as shown in table I.1) are intended to be used in the EIT when the broadcaster wants to indicate the video characteristics for a particular event are less than the capabilities of the service indicated by the *service_type* and component_descriptor in the SDT.

**Table I.18: Component types from table 26 to inform on the source's spatial,
temporal, and dynamic range characteristics**

| *stream_content* | *stream_content_ext* | *component_type* | Description |
|---|---|---|---|
| 0x3 | n/a | 0x40 | Video spatial resolution has been upscaled from lower resolution source material |
| | | 0x41 | Video is SDR |
| | | 0x42 | Video is HDR remapped from SDR source material |
| | | 0x43 | Video is HDR up-converted from SDR source material |
| | | 0x44 | Video is standard frame rate, less than or equal to 60 Hz |
| | | 0x45 | High frame rate video generated from lower frame rate source material |
| 0xB | 0xF | 0x04 | HLG10 HDR |
| | | 0x06 | SMPTE ST 2094-10 DMI format as defined in ETSI TS 101 154 [14] |
| | | 0x07 | SL-HDR2 DMI format as defined in ETSI TS 101 154 [14] |
| | | 0x08 | SMPTE ST 2094-40 DMI format as defined in ETSI TS 101 154 [14] |
| | | 0x09 | PQ10 HDR |

## I.2.9.3 Examples of signalling for different bitstream profiles using service_type 0x22

**Table I.19: Signalling for AVS3 bitstream profiles**

| AVS3 profile | *service_type* | Mandatory SDT component_descriptor | | | Optional component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | *stream_content* | *stream_content_ext* | *component_type* | *stream_content* | *stream_content_ext* | *component_type* |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) standard dynamic range frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x20 | 0x3 | n/a | 0x41 |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range HLG frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x20 | 0xB | 0xF | 0x04 |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x20 | 0xB | 0xF | 0x09 |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-1" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI | 0x22 | 0x9 | 0x0 | 0x20 | 0xB | 0xF | 0x06 |
| | | | | | 0xB | 0xF | 0x07 |
| | | | | | 0xB | 0xF | 0x08 |

| AVS3 profile | service_type | Mandatory SDT component_descriptor | | | Optional component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | stream_content | stream_content_ext | component_type | stream_content | stream_content_ext | component_type |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) standard dynamic range frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x21 | 0x3 | n/a | 0x41 |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range HLG frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x21 | 0xB | 0xF | 0x04 |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x21 | 0xB | 0xF | 0x09 |
| AVS3 bitstream with resolution up to 3 840 x 2 160 (including 1 920 x 1 080) high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-1" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI | 0x22 | 0x9 | 0x0 | 0x21 | 0xB<br>0xB<br>0xB | 0xF<br>0xF<br>0xF | 0x06<br>0x07<br>0x08 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 standard dynamic range frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x22 | 0x3 | n/a | 0x41 |

| AVS3 profile | service_type | Mandatory SDT component_descriptor | | | Optional component_descriptor (see note 1) | | |
|---|---|---|---|---|---|---|---|
| | | stream_content | stream_content_ext | component_type | stream_content | stream_content_ext | component_type |
| AVS3 bitstream with resolution up to 7 680 x 4 320 high dynamic range HLG frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x22 | 0xB | 0xF | 0x04 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x22 | 0xB | 0xF | 0x09 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 60 Hz conformant to the conformance point "AVS3 HDR UHDTV-2" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI | 0x22 | 0x9 | 0x0 | 0x22 | 0xB 0xB 0xB | 0xF 0xF 0xF | 0x06 0x07 0x08 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 standard dynamic range frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x23 | 0x3 | n/a | 0x41 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 high dynamic range HLG frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x23 | 0xB | 0xF | 0x04 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] | 0x22 | 0x9 | 0x0 | 0x23 | 0xB | 0xF | 0x09 |
| AVS3 bitstream with resolution up to 7 680 x 4 320 high dynamic range PQ frame rate up to 120 Hz conformant to the conformance point "AVS3 HDR HFR UHDTV-2" as defined in ETSI TS 101 154 [14] and containing SEI messages carrying DMI | 0x22 | 0x9 | 0x0 | 0x23 | 0xB 0xB 0xB | 0xF 0xF 0xF | 0x06 0x07 0x08 |
| NOTE: These component_descriptors are optional to broadcast. A broadcaster can signal them if they wish for receivers to be able to group services by their AVS3 bit stream profile. There are no requirements for receivers to use these. | | | | | | | |

# Annex J (normative):
# Signalling of supplementary audio

## J.1    Overview

The `supplementary_audio_descriptor` provides additional information about audio streams to allow a receiver to present the audio to the user in the appropriate way. The `supplementary_audio_descriptor` can also be used to describe non-supplementary audio components.

The `supplementary_audio_descriptor` can override the language code in the `ISO_639_language_descriptor` (ISO/IEC 13818-1 [1]) or other descriptors in the same *ES_info* loop of a PMT. This is commonly used where the language code in the `ISO_639_language_descriptor` is set to aid legacy receivers. In such cases the more accurate language code is conveyed in the `supplementary_audio_descriptor`.

For legacy receivers, the `ISO_639_language_descriptor` may also continue to be carried.

Three mechanisms are defined for the provision of supplementary audio (for example audio description):

- receiver-mix,

- broadcast-mix, and

- parametric data stream.

In the first two cases, the audio exists as two separate audio recordings, one recording is a complete and independent audio, the other is an additional audio recording, that will only be listened to by the viewer when mixed with the complete and independent audio. A broadcaster then broadcasts a complete and independent audio, and in addition one or both of the following two options:

1) The broadcaster transmits an extra audio component which is a mix of the complete and independent audio, and the additional audio recording. This is called broadcast-mix audio.

2) The broadcaster transmits the additional audio recording as an extra audio component (known as dependent audio) because it is intended to be mixed with a complete and independent audio stream by the receiver. This is called receiver-mix audio.

SAOC-DE parametric data streams are described further in clause J.5.

Annex J defines the signalling in each case to enable the receiver to present the correct audio to the user.

## J.2    Receiver-mix supplementary audio

## J.2.1    Introduction

Receiver-mix audio allows receivers to give the listener control over the relative levels of the mix, to improve intelligibility. Where it is important to conserve bit rate, receiver-mix supplementary audio offers the broadcaster a way to provide a useful feature for users without having to broadcast another complete audio channel. However, it requires the receiver to implement two audio decoders and a mixer.

Annex E of ETSI TS 101 154 [14] defines additional signalling which controls how the receiver mixes the complete and independent audio stream with the dependent audio stream before presentation to the user.

Receivers shall use attributes such as *ISO_639_language_code* and *editorial_classification* to determine the complete and independent audio stream to mix with a dependent audio stream.

Receiver mixing of audio streams shall apply the following restrictions, which depend on values signalled in the `supplementary_audio_descriptor`:

- A dependent stream (*mix_type* 0) shall only be mixed with a complete and independent stream (*mix_type* 1).

- A dependent stream with audio description for the visually impaired (*editorial_classification* 0x01) shall only be mixed with a complete and independent stream associated with the same language code. For other editorial classifications, the dependent, and complete and independent streams may use different language codes.

- An audio stream with an *editorial_classification* in the range 0x00 to 0x16 (inclusive) shall not be mixed with another stream which has the same editorial classification as itself, regardless of whether it is either a dependent, or a complete and independent stream.

EXAMPLE:        It is, for example, hence not allowed to mix a broadcast-mix spoken subtitles stream with a dependent spoken subtitles stream.

Note that ETSI TS 101 154 [14] specifies that in the case of receiver-mix, the main audio and the dependent audio shall be encoded using the same codec family.

# J.2.2    PSI PMT signalling

When a receiver-mix supplementary dependent stream is present, it shall be signalled in the *ES_info* descriptor loop of the PSI PMT with both an *ISO_639_language_code* in the `ISO_639_language_descriptor` (ISO/IEC 13818-1 [1]) set to reflect the content of the audio stream, and a `supplementary_audio_descriptor`. The `supplementary_audio_descriptor` shall set the *mix_type* to 0 and the *editorial_classification* to any appropriate value other than 0. The other fields shall be set according to clause J.4.

A `stream_identifier_descriptor` should be present.

# J.2.3    EIT signalling

## J.2.3.1    General principles

A `component_descriptor`, with the same value of *component_tag* as that used in the PSI PMT `stream_identifier_descriptor` of the dependent audio stream, should be present in the descriptor loop in the EIT for each event for which supplementary audio is broadcast.

## J.2.3.2    Visually impaired audio description

As defined in table 26, for visually impaired audio description the values of *stream_content*/*stream_content_ext*/*component_type* given in table J.1 shall be used.

**Table J.1: Receiver-mix component descriptor values for visually impaired audio description**

| Audio coding | stream_content | stream_content_ext | component_type |
|---|---|---|---|
| MPEG-1 Layer 2 | 0x2 | 0xF | 0x47 |
| Enhanced AC-3 | 0x4 | 0xF | 0x90 (see note 1) |
| AC-4 | 0x9 | 0x1 | 0x0C to 0x0D (see note 2) |
| HE-AAC, AAC (see note 3) | 0x6 | 0xF | 0x47 |
| HE-AAC v2 | 0x6 | 0xF | 0x49 |
| DTS-HD | 0x7 | 0xF | 0bx001 0xxx (see note 4) |
| DTS-UHD | 0x9 | 0x1 | 0x1C to 0x1D (see note 5) |
| NOTE 1:   As defined in annex D.<br>NOTE 2:   The *component_type* value (defined in clause 6.2.8) is dependent upon the AC-4 coding parameters.<br>NOTE 3:   The AAC profile (which includes low-complexity AAC) also uses this type (see clause 6.2.8).<br>NOTE 4:   As defined in clause G.3.4.<br>NOTE 5:   The *component_type* value (defined in clause 6.2.8) is dependent upon the DTS-UHD coding parameters. | | | |

# J.3        Broadcast-mix supplementary audio

## J.3.1        Introduction

Broadcast-mix supplementary audio may be used in broadcast systems where bit rate is available for a separate audio stream or where compatibility with legacy receivers is important. It requires no special handling in the audio decoder of the receiver, however receivers should recognize that this audio stream is different from the main programme audio stream using the information in the `ISO_639_language_descriptor` and `supplementary_audio_descriptor`.

## J.3.2        PSI PMT signalling

When a broadcast-mix supplementary audio stream is present, it shall be signalled in the PSI PMT with a `supplementary_audio_descriptor`. The *ISO_639_language_code* field in the `ISO_639_language_descriptor` (ISO/IEC 13818-1 [1]) should be set to reflect the content of the audio stream, however this may be an issue with legacy receivers which do not interpret the `supplementary_audio_descriptor`. For this reason, some broadcasters may decide to use a different language code in the *ISO_639_language_code* field (e.g. "NAR" or "qad") in which case the correct language code shall be signalled in the `supplementary_audio_descriptor`. The other fields shall be set according to clause J.4.

A `stream_identifier_descriptor` should be present.

## J.3.3        EIT signalling

### J.3.3.1        General principles

A `component_descriptor`, with the same value of *component_tag* as that used in the PSI PMT `stream_identifier_descriptor` of the supplementary audio stream, should be present in the descriptor loop in the EIT for each event for which a supplementary audio stream is broadcast.

### J.3.3.2        Visually impaired audio description

As defined in table 26, for visually impaired audio description the values of *stream_content*/*stream_content_ext*/*component_type* given in table J.2 shall be used.

**Table J.2: Broadcast-mix component descriptor values for visually impaired audio description**

| Audio coding | stream_content | stream_content_ext | component_type |
|---|---|---|---|
| MPEG-1 Layer 2 | 0x2 | 0xF | 0x48 |
| AC-3 | 0x4 | 0xF | 0b0101 0xxx (see note 1) |
| Enhanced AC-3 | 0x4 | 0xF | 0b1101 0xxx (see note 1) |
| AC-4 | 0x9 | 0x1 | 0x06 to 0x0B (see note 2) |
| HE-AAC, AAC (see note 3) | 0x6 | 0xF | 0x48 |
| HE-AAC v2 | 0x6 | 0xF | 0x4A |
| DTS | 0x7 | 0xF | 0bx101 0xxx (see note 4) |
| DTS-HD | 0x7 | 0xF | 0bx101 0xxx (see note 4) |
| DTS-UHD | 0x9 | 0x1 | 0x16 to 0x1B (see note 5) |
| NOTE 1:  "xxx" is dependent upon the number of channels, as defined in annex D. | | | |
| NOTE 2:  The *component_type* value (defined in clause 6.2.8) is dependent upon the AC-4 coding parameters. | | | |
| NOTE 3:  The AAC profile (which includes low-complexity AAC) also uses this type (see clause 6.2.8). | | | |
| NOTE 4:  "xxx" is dependent upon the number of channels, as defined in clause G.3.4. | | | |
| NOTE 5:  The *component_type* value (defined in clause 6.2.8) is dependent upon the DTS-UHD coding parameters. | | | |

# J.4    PSI signalling of audio purpose

Table J.3 lists the combination of valid parameters that shall be signalled in the PMT. All other combinations are invalid and shall not be used.

**Table J.3: PMT field values for different audio purposes**

| Audio purpose | *audio_type* (see note 1) | *mix_type* (see note 2) | *editorial_classification* (see note 2) |
|---|---|---|---|
| Main audio (see note 3) | 0x00 or 0x01 | 1 | 0x00 |
| Audio description (broadcast-mix) | 0x00, 0x01, or 0x03 | 1 | 0x01 |
| Audio description (receiver-mix) | 0x03 | 0 | 0x01 |
| Clean audio (broadcast-mix) | 0x02 | 1 | 0x02 |
| Parametric data dependent stream (see note 4) | 0x02 | 0 | 0x04 |
| Spoken subtitles (broadcast-mix) | 0x00, 0x01, or 0x03 | 1 | 0x03 |
| Spoken subtitles (receiver-mix) | 0x03 | 0 | 0x03 |
| Unspecific audio for the general audience | any | 0 or 1 | 0x17 |
| user defined | any | 0 or 1 | 0x18 to 0x1F |
| NOTE 1: *audio_type* is broadcast in the `ISO_639_language_descriptor` (ISO/IEC 13818-1 [1]). | | | |
| NOTE 2: *mix_type* and *editorial_classification* are broadcast in the `supplementary_audio_descriptor` (see clause 6.4.11). | | | |
| NOTE 3: The `supplementary_audio_descriptor` is optional in this case. | | | |
| NOTE 4: Embedded parametric data shall not be signalled with *editorial_classification* 0x04. | | | |

# J.5    SAOC-DE parametric data streams

## J.5.1    Introduction

The SAOC-DE parametric data is usually embedded as ancillary data in the audio stream. Alternatively, if the audio codec does not provide such capabilities, the SAOC-DE parametric data may also be conveyed as a separate dependent data stream on a separate component.

When the SAOC-DE parametric data is conveyed in a separate elementary stream, receivers shall use attributes such as *ISO_639_language_code* and *editorial_classification* to determine which complete and independent audio stream to combine with the dependent SAOC-DE parametric data stream.

Receivers shall apply the following restrictions when combining an audio and a dependent SAOC-DE parametric data stream. The restrictions depend on values signalled in the `supplementary_audio_descriptor`:

- A dependent SAOC-DE parametric data stream shall be signalled as *mix_type* 0 and shall only be combined with a complete and independent audio stream (*mix_type* 1).

- A dependent SAOC-DE parametric data stream shall only be combined with a complete and independent stream associated with the same language code.

## J.5.2    PSI PMT signalling

When a dependent SAOC-DE parametric data stream is present, it shall be signalled in the *ES_info* descriptor loop of the PSI PMT with both an *ISO_639_language_code* in the `ISO_639_language_descriptor` (ISO/IEC 13818-1 [1]) set to reflect the content of the audio stream, and a `supplementary_audio_descriptor`. The *audio_type* in the `ISO_639_language_descriptor` shall be set to 0x2. The `supplementary_audio_descriptor` shall set the *mix_type* to 0, and the *editorial_classification* to 0x4.

When the SAOC-DE parametric data is embedded as ancillary data in an AAC audio stream, the presence of embedded SAOC-DE shall be signalled using the `AAC_descriptor` defined in clause H.2.1. The `ISO_639_language_descriptor` (ISO/IEC 13818-1 [1]) and an optional `supplementary_audio_descriptor` shall be used to reflect the content of the audio stream, but set as if the SAOC-DE parametric data was not present. A `supplementary_audio_descriptor` with *editorial_classification* 0x4 shall not be used for audio components with embedded SAOC-DE parametric data.

A `stream_identifier_descriptor` should be present for both dependent, and embedded SAOC-DE parametric data elementary streams.

# J.5.3    EIT signalling

When a dependent SAOC-DE parametric data stream is present, a `component_descriptor` (*stream_content* 0x3, *stream_content_ext* 0xF, *component_type* 0x80) with the same value of *component_tag* as that used in the PSI PMT `stream_identifier_descriptor` of the dependent SAOC-DE parametric data stream, should be present in the descriptor loop in the EIT for each event for which the dependent SAOC-DE data stream is broadcast.

When the SAOC-DE parametric data is embedded as ancillary data in an AAC audio stream, a `component_descriptor` (*stream_content* 0x6, *stream_content_ext* 0xF, *component_type* 0xA0) with the same value of *component_tag* as that used in the PSI PMT `stream_identifier_descriptor` for the audio stream, should be present in the descriptor loop in the EIT for each event for which the SAOC-DE data stream is broadcast. Additional `component_descriptor` with the same value of *component_tag* should also be present to indicate the type of the AAC audio stream itself.

These recommendations are summarized in table J.4.

**Table J.4: Use of the component_descriptor for SAOC-DE parametric data streams**

| Audio coding | stream_content | stream_content_ext | component_type |
|---|---|---|---|
| Dependent SAOC-DE data stream | 0x3 | 0xF | 0x80 |
| HE-AAC, HE-AAC v2, or AAC (see note) with SAOC-DE ancillary data | 0x6 | 0xF | 0xA0 |
| NOTE:    AAC also uses this type (see note 7 in table 26 in clause 6.2.8). | | | |

# Annex K (normative):
# Use of the extended_event_linkage_info

The present annex is to explain the valid signalling combinations for the `linkage_descriptor` carrying `extended_event_linkage_info` which is defined in clause 6.2.19.4. Other signalling combinations not specified in the present annex are reserved for future use. It is expected the present annex will expand as use cases for new signalling combinations become evident.

The `linkage_descriptor` carrying `extended_event_linkage_info` allows a broadcaster to signal linkages between EPG events in the same or different services. A `linkage_descriptor` carrying `extended_event_linkage_info` is added to the EIT of the "source" event and signals a linkage to one or more "destination" events.

As explained in annex I, the exact type of a service cannot be determined using the *service_type* alone. So in the following valid use-cases, the description of the type of service (during the event) is described generically rather than by *service_type*.

Linkages from an event to itself shall not be used, since it is possible for receivers to determine the same information from the PSI and SI without the need for a `linkage_descriptor` carrying `extended_event_linkage_info`.

For certain use cases it may be applicable to link from a "source" event to a single "destination" event multiple times, each with a different combination of *linkage_type* and *link_type*.

The valid signalling combinations of the `linkage_descriptor` carrying `extended_event_linkage_info` are shown in table K.1.

**Table K.1: Valid use-cases of extended_event_linkage_info**

| Type of source event | Type of destination event | *linkage_type* (see note 1) | *link_type* (see note 2) | Use-case |
|---|---|---|---|---|
| SD | SD | 0xE | 0 | Link to alternate event instances also in SD |
| SD | HD | 0xE | 1 | Link to event in HD |
| SD | Frame Compatible (FC)-3DTV | 0xE | 2 | Link to event in frame compatible plano-stereoscopic |
| SD | Service Compatible (SC)-3DTV MVC | 0xE | 1 | Link to event in service compatible plano-stereoscopic MVC which provides an HD alternative (see note 3) |
| SD | SC-3DTV MVC | 0xE | 3 | Link to event in service compatible plano-stereoscopic MVC |
| SD | UHD | 0xF | 0 | Link to event in UHD |
| SD | SFC-3DTV HEVC | 0xF | 1 | Link to event in service frame compatible plano-stereoscopic |
| HD | SD | 0xE | 0 | Link to event in SD |
| HD | HD | 0xE | 1 | Link to alternate event instances also in HD |
| HD | FC-3DTV | 0xE | 2 | Link to event in frame compatible plano-stereoscopic |
| HD | SC-3DTV MVC | 0xE | 3 | Link to event in service compatible plano-stereoscopic MVC |
| HD | UHD | 0xF | 0 | Link to event in UHD |
| HD | SFC-3DTV HEVC | 0xF | 1 | Link to event in service frame compatible plano-stereoscopic |
| FC-3DTV | SD | 0xE | 0 | Link to event in SD |
| FC-3DTV | HD | 0xE | 1 | Link to event in HD |
| FC-3DTV | FC-3DTV | 0xE | 2 | Link to alternate event instances also in frame compatible plano-stereoscopic |
| FC-3DTV | SC-3DTV MVC | 0xE | 1 | Link to event in service compatible plano-stereoscopic MVC which provides an HD alternative (see note 3) |
| FC-3DTV | SC-3DTV MVC | 0xE | 3 | Link to event in service compatible plano-stereoscopic MVC |
| FC-3DTV | UHD | 0xF | 0 | Link to event in UHD |
| FC-3DTV | SFC-3DTV HEVC | 0xF | 1 | Link to event in service frame compatible plano-stereoscopic |

| Type of source event | Type of destination event | *linkage_type* (see note 1) | *link_type* (see note 2) | Use-case |
|---|---|---|---|---|
| SC-3DTV MVC | SD | 0xE | 0 | Link to event in SD |
| SC-3DTV MVC | HD | 0xE | 1 | Link to event in HD |
| SC-3DTV MVC | FC-3DTV | 0xE | 2 | Link to event in frame compatible plano-stereoscopic |
| SC-3DTV MVC | SC-3DTV MVC | 0xE | 1 | Link to event in service compatible plano-stereoscopic MVC which provides a better HD alternative (see note 3) |
| SC-3DTV MVC | SC-3DTV MVC | 0xE | 3 | Link to alternate event instances also in service compatible plano-stereoscopic MVC |
| SC-3DTV MVC | UHD | 0xF | 0 | Link to event in UHD |
| SC-3DTV MVC | SFC-3DTV HEVC | 0xF | 1 | Link to event in service frame compatible plano-stereoscopic |
| SFC-3DTV HEVC | SD | 0xE | 0 | Link to event in SD |
| SFC-3DTV HEVC | HD | 0xE | 1 | Link to event in HD |
| SFC-3DTV HEVC | FC-3DTV | 0xE | 2 | Link to event in frame compatible plano-stereoscopic |
| SFC-3DTV HEVC | SC-3DTV MVC | 0xE | 1 | Link to event in service compatible plano-stereoscopic MVC which provides a better HD alternative (see note 3) |
| SFC-3DTV HEVC | SC-3DTV MVC | 0xE | 3 | Link to event in service compatible plano-stereoscopic MVC |
| SFC-3DTV HEVC | UHD | 0xF | 0 | Link to event in UHD |
| SFC-3DTV HEVC | SFC-3DTV HEVC | 0xF | 1 | Link to alternate event instances also in service frame compatible plano-stereoscopic |
| UHD | SD | 0xE | 0 | Link to event in SD |
| UHD | HD | 0xE | 1 | Link to event in HD |
| UHD | FC-3DTV | 0xE | 2 | Link to event in frame compatible plano-stereoscopic |
| UHD | SC-3DTV MVC | 0xE | 3 | Link to event in service compatible plano-stereoscopic MVC |
| UHD | UHD | 0xF | 0 | Link to alternate event instances also in UHD |
| UHD | SFC-3DTV HEVC | 0xF | 1 | Link to event in service frame compatible plano-stereoscopic |
| NOTE 1: See table 60. | | | | |
| NOTE 2: See table 66. | | | | |
| NOTE 3: Only use if destination event is the best HD alternative. | | | | |

# Annex L (normative):
# Service information implementation of DTS Neural™ Surround coded audio in DVB systems

## L.0      Introduction

DTS Neural™ Surround constitutes a method of encoding and decoding additional audio channels into a host audio stream. A common application example would be to encode 5.1 channel surround sound into a stereo audio program.

DTS Neural™ Surround coding is an active intensity phase encoding methodology, and does not carry embedded metadata signalling. A system unaware of this processing will operate without impediment. The nature of this encoding permits the carriage independent of audio encoding algorithm, provided sufficient quality is maintained.

A common system application example would be to use DTS Neural™ Surround to encode 5.1 channel surround sound for a stereo MPEG 1 layer 2 host audio, enabling the delivery of surround sound audio in early generations of digital media distribution systems which were designed for stereo audio distribution.

The transmission of DTS Neural™ Surround is optional in DVB systems, as specified in ETSI TS 101 154 [14]. However, if DTS Neural™ Surround is transmitted in a DVB system the specifications in annex L shall be followed.

## L.1      DTS Neural descriptor

The `DTS_Neural_descriptor` is used in the PMT to identify which audio streams have been processed with DTS Neural™ Surround encoding. In such cases, a `DTS_Neural_descriptor` follows the associated audio descriptor in the `program_map_section` *ES_info* descriptor loop. The `DTS_Neural_descriptor` is described in table L.1.

**Table L.1: DTS Neural descriptor**

| Syntax | Number of bits | Identifier |
|---|---|---|
| `DTS_Neural_descriptor() {` | | |
|    `descriptor_tag` | 8 | uimsbf |
|    `descriptor_length` | 8 | uimsbf |
|    `descriptor_tag_extension` | 8 | uimsbf |
|    `config_id` | 8 | uimsbf |
|    `for (i=0;i<N;i++) {` | | |
|       `additional_info_byte` | 8 | uimsbf |
|    `}` | | |
| `}` | | |

Semantics for the DTS Neural descriptor:

*config_id***:** This 8-bit field indicates the audio channel configuration of the host audio stream. For a stereo host audio stream it is coded according to table L.2, and for a 5.1 host audio stream it is coded according to table L.3.

**Table L.2: Audio channel configuration for DTS Neural stereo host audio stream**

| config_id | Original audio configuration (see note 1) | Original channel count (see note 2) |
|---|---|---|
| 0 | unknown or undefined | |
| 1 | L, R | 2 |
| 2 | L, R, C | 3 |
| 3 | L, R, Ls, Rs | 4 |
| 4 | L, R, C, Ls, Rs | 5 |
| 5 | L, R, C, Ls, Rs, Cs | 6 |
| 6 | L, R, C, Ls, Rs, Lb, Rb | 7 |
| 7 | L, R, Ls, Rs, Cs | 5 |
| 8 | L, R, Ls, Rs, Lb, Rb | 6 |
| 9 to 255 | reserved for future use | |
| NOTE 1:  L = left, R = right, C = centre, s = surround, b = back. | | |
| NOTE 2:  The LFE channel is omitted from the original audio configuration for stereo host audio streams. If an LFE channel existed in the original audio configuration, that channel was mixed into the full band channels during DTS Neural™ Surround encoding. An LFE channel can be generated by a bass management system. | | |

**Table L.3: Audio channel configuration for DTS Neural 5.1 host audio stream**

| config_id | Original audio configuration (see note) | Original channel count |
|---|---|---|
| 0 | unknown or undefined | |
| 1 | L, R, C, LFE, Ls, Rs | 5.1 |
| 2 | L, R, C, LFE, Ls, Rs, Cs | 6.1 |
| 3 | L, R, C, LFE, Ls, Rs, Lb, Rb | 7.1 |
| 4 to 255 | reserved for future use | |
| NOTE:    L = left, R = right, C = centre, LFE = low frequency effects, s = surround, b = back. | | |

*additional_info_byte*: This 8-bit field is optional and reserves a place for future expansion of this descriptor.

# Annex M (normative):
# Signalling of next-generation audio

## M.1    Overview

NGA systems provide broadcasters, operators, and content providers with more flexibility to create and deliver their content. These NGA systems introduce a number of new concepts and techniques, including:

- the capability to provide immersive audio with the addition of height elements;

- the capability to provide personalization options to broadcasters and consumers;

- the introduction of audio objects to facilitate immersive and personalized audio.

The usage of multiple audio preselections allows broadcasters, operators, and content providers to offer different variations of the content from a single NGA audio programme. A more detailed explanation of these concepts and techniques is given in annex K of ETSI TS 101 154 [14].

The `audio_preselection_descriptor` is used for signalling the NGA features that the receiver may use for selection of the appropriate audio preselection to present the audio programme to the user.

For each signalled audio preselection, the descriptor provides information about the available accessibility features and languages, indication about the preferred reproduction layout and the interactivity options.

## M.2    PSI PMT signalling

If the `audio_preselection_descriptor` is used for one audio programme contained in one or more NGA elementary streams, neither the `ISO_639_language_descriptor` nor the `supplementary_audio_descriptor` should be present in the *ES_info* descriptor loop of any of the NGA streams (main or auxiliary) of an audio programme.

When an `audio_preselection_descriptor` is present in the *ES_info* loop for an NGA elementary stream in the PMT, receivers shall ignore any `supplementary_audio_descriptor` or `ISO_639_language_descriptor` which are present in the same *ES_info* loop.

## M.3    Mapping of codec-specific values to the audio preselection descriptor (informative)

Table M.1 lists for each NGA codec the mapping between the codec-specific values and the fields present in the `audio_preselection_descriptor`.

**Table M.1: Mapping of codec-specific values to the audio preselection descriptor**

| NGA codec | Field in the `audio_preselection_descriptor` | Mapping |
|---|---|---|
| AC-4 Part-2 | *num_preselections* | *num_presentation* field (within the AC-4 TOC according to clause 6.7 of ETSI TS 101 154 [14]). |
| AC-4 Part-2 | *preselection_id* | *presentation_group_index* field of the *Preselection* (within the AC-4 TOC according to clause 6.7 of ETSI TS 101 154 [14]). |
| AC-4 Part-2 | *ISO_639_language_code* | The first language indicated by this field should correspond to the information conveyed in the *language_tag_bytes* field (within the AC-4 TOC according to clause 6.7 of ETSI TS 101 154 [14]) for the presentation, that is tagged as dialog or complete main. |

| NGA codec | Field in the `audio_preselection_descriptor` | Mapping |
|---|---|---|
| AC-4 Part-2 | *audio_description* | This field should correspond to the information conveyed in the *content_classifier* field of the `ac4_presentation_info` structure (within the AC-4 TOC according to clause 6.7 of ETSI TS 101 154 [14]) for the presentation where at least one substream is tagged as Visually Impaired (VI). |
| AC-4 Part-2 | *spoken_subtitles* | This field should correspond to the information conveyed in the *content_classifier* field of the `ac4_presentation_info` structure (within the AC-4 TOC according to clause 6.7 of ETSI TS 101 154 [14]) for the presentation where at least one substream is tagged as Voice Over (VO). |
| AC-4 Part-2 | *dialogue_enhancement* | This field should correspond to the information conveyed in the *content_classifier* field of the `ac4_presentation_info` structure (within the AC-4 TOC according to clause 6.7 of ETSI TS 101 154 [14]) for the presentation where at least one substream is tagged as Hearing Impaired (HI) or Dialog (D). |
| MPEG-H | *num_preselections* | *mae_numGroupPresets* field as specified in clause 6.8 of ETSI TS 101 154 [14]. |
| MPEG-H | *preselection_id* | *mae_GroupPresetID* field as specified in clause 6.8 of ETSI TS 101 154 [14]. |
| MPEG-H | *ISO_639_language_code* | The first language indicated by this field should correspond to the information conveyed in *mae_contentLanguage* of the default dialog element: the *maeGroup* which is marked as default in *mae_switchGroupDefaultGroupID* and is tagged in *mae_contentKind* as dialog. This information is carried in the `AudioSceneInformation()` of the MPEG-H audio stream according to clause 6.8 of ETSI TS 101 154 [14]. |
| MPEG-H | *audio_description* | This field should correspond to the *mae_groupPresetKind* value in the `mae_GroupPresetDefinition()` structure and the *mae_contentKind* values in the `mae_ContentData()` structures in the `AudioSceneInformation()` of the MPEG-H audio stream according to clause 6.8 of ETSI TS 101 154 [14]. A value of 1 in *audio_description* field corresponds to value 7 in *mae_groupPresetKind* and value 9 in *mae_contentKind*. |
| MPEG-H | *spoken_subtitles* | This field should correspond to the *mae_groupPresetKind* value in the `mae_GroupPresetDefinition()` structure and the *mae_contentKind* values in the `mae_ContentData()` structures in the `AudioSceneInformation()` of the MPEG-H audio stream according to clause 6.8 of ETSI TS 101 154 [14]. A value of 1 in *spoken_subtitles* field corresponds to value 8 in *mae_groupPresetKind* and value 8 in *mae_contentKind*. |
| MPEG-H | *dialogue_enhancement* | This field should correspond to the *mae_groupPresetKind* value in the `mae_GroupPresetDefinition()` structure and the *mae_contentKind* values in the `mae_ContentData()` structures in the `AudioSceneInformation()` of the MPEG-H Audio stream according to clause 6.8 of ETSI TS 101 154 [14]. A value of 1 in *dialogue_enhancement* field corresponds to value 2 in *mae_contentKind*. |
| DTS-UHD | *num_preselections* | The number of preselections available in the audio programme, as constrained by clause 6.4.1 and indicated in the *BroadcastChunk* as described in clause G.5.2.1. |
| DTS-UHD | *preselection_id* | The *preselection_id* is related to the *ProgramIndex* in the *BroadcastChunk*, using the equation shown in clauseG.5.2.1 for the associated language. |
| DTS-UHD | *ISO_639_language_code* | If this value is present, it should equal *ISO639_code* in the *BroadcastChunk* for the corresponding *preselection_id* as described in clause 6.9.3.1 of ETSI TS 101 154 [14]. |
| DTS-UHD | *audio_description* | The value of *audio_description* should correspond to that of the *AudioDescription* field in the corresponding *SelectionSet* in the *BroadcastChunk* indicated by *preselection_id* as described in clause 6.9.3.1 of ETSI TS 101 154 [14]. |
| DTS-UHD | *spoken_subtitles* | The value of *spoken_subtitles* should correspond to that of the *SpokenSubtitles* field in the corresponding *SelectionSet* in the *BroadcastChunk* indicated by *preselection_id* as described in clause 6.9.3.1 of ETSI TS 101 154 [14]. |

| NGA codec | Field in the `audio_preselection_descriptor` | Mapping |
|---|---|---|
| DTS-UHD | *dialogue_enhancement* | The value of *dialogue_enhancement* should correspond to that of the *DialogueEnhancement* field in the corresponding *SelectionSet* in the *BroadcastChunk* indicated by *preselection_id* as described in clause 6.9.3.1 of ETSI TS 101 154 [14]. |

# Annex N (informative):
# Examples for using multiple component descriptors

Some `component_descriptor` values in table 26 are marked as being intended to be present in combination with one or more component descriptors with the same *component_tag* value. All component descriptors using the same *component_tag* value, and which are part of the same service or event description (depending on the location of the component descriptors), describe one and the same component. This means that the information about a component being conveyed in SI, is the combination of the information provided by all component descriptors referring to this service or event component:

- For example, two component descriptors with the same *component_tag* value, and *stream_content*/*stream_content_ext*/*component_type* values of 0x5/0xF/0x0B and 0x3/0xF/0x40 respectively, would indicate H.264/AVC high definition video, 16:9 aspect ratio, 25 Hz that has been up-sampled from a standard definition source.

- For example, two component descriptors with the same *component_tag* value, and *stream_content*/*stream_content_ext*/*component_type* values of 0x6/0xF/0x03 and 0x6/0xF/0xA0 respectively, would indicate stereo HE-AAC audio with SAOC-DE parametric data embedded as ancillary data.

- For example, two component descriptors with the same *component_tag* value, and *stream_content*/*stream_content_ext*/*component_type* values 0x9/0x0/0x01 and 0xB/0xF/0x01 respectively, would indicate HEVC Main 10 profile high definition video at 50 Hz with a 16:9 aspect ratio.

- For example, two components which together constitute a video stream of HLG10 HDR, remapped from SDR source material, frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz, or 120 Hz with half frame rate HEVC temporal video sub-bitstream. The main component which is decodable by HEVC_UHDTV_IRD would be signalled using three component descriptors with *stream_content*/*stream_content_ext*/*component_type* values 0x9/0x0/0x04, 0x3/0xF/0x42 and 0xB/0xF/0x04 respectively. The other component would be signalled using *stream_content*/*stream_content_ext*/*component_type* value 0xB/0xF/0x05 would indicate HEVC temporal video subset for a frame rate of 100 Hz, $\frac{120\,000}{1\,001}$ Hz or 120 Hz. There is no need to include component descriptors with *stream_content*/*stream_content_ext*/*component_type* values 0x9/0x0/0x04, 0x3/0xF/0x42 for the secondary component since this information is implied from signalling of the main component.

- For example, four component descriptors with the same *component_tag* value, with one `component_descriptor` having a *stream_content*/*stream_content_ext*/*component_type* value of 0x9/0x0/0x05 (indicating HEVC ultra high definition video with PQ10 HDR) and the other three component descriptors indicating the presence of three kinds of DMI in the video stream. The three component descriptors indicating DMI have *stream_content*/*stream_content_ext*/*component_type* values of 0xB/0xF/0x06, 0xB/0xF/0x07 and 0xB/0xF/0x08 respectively.

# Annex O (informative):
# Bibliography

- IEC 62106: "Specification of the radio data system (RDS) for VHF/FM sound broadcasting in the frequency range from 87,5 to 108,0 MHz".

- RDS-Forum SPB 490: "RDS Universal Encoder Communication Protocol", Final Version 6.01, June 2003.

- ETSI TS 102 584: "Digital Video Broadcasting (DVB); DVB-SH implementation guidelines".

# Annex P (informative):
# Change History

| Date | Version | Information about changes |
|---|---|---|
| June 2015 | 1.15.1 | Implemented Change Requests: <br> • 847: 3DTV phase 3 signalling <br> • 953: new URI linkage type for MRS identifier <br> • 986: expanding the capability of the AAC Descriptor to support clean_audio <br> • 1008: Allocate bit in adaptation field_data_identifier for TSAP timeline <br> • 1031: MVC dependent view component descriptor (note about 24 Hz is confusing) <br> • 1035: annex A code point differences since revision 1.12.1 <br> • 1041: a reserved field should be reserved for future use in supplementary audio descriptor <br> • 1093: service type 0x1F for HEVC digital television service <br> • 1096: update annex I for HEVC <br> • 1098: stream content and component type for HEVC <br> • 1142: signal presence of multi-region disparity SEI data <br> • 1156: naming of HEVC UHD component type? <br> • 1181: promote various audio annexes from informative to normative <br> • 1203: various typos in r99 of draft of ETSI EN 300 468 <br> • 1204: explain use of component descriptor for service list filtering <br> • 1205: add signalling for Dolby AC-4 <br> • 1298: note 7 in component descriptor table 26 in clause 6.2.8 <br> • 1310: note 4 in annex J tables J.1 and J.2 confusing. <br> • 1311: clause I.2.5 add additional text to warn explicit about future backwards compatible services <br> • 1313: table K.1 in annex K needs updating to include new SFC-3DTV event linkage value <br> • 1317: update C2 delivery system descriptor for PLP bundling <br> • 1320: add a revision history table at the end <br> • 1326: descriptor_tag_extension 0x16 for C2 bundle delivery system descriptor |

| Date | Version | Information about changes |
|------|---------|---------------------------|
| June 2017 | 1.16.1 | Implemented Change Requests:<br>• 671: confusion wrt. bit order in bitmapped fields<br>• 944: ETSI reference for TM-CSS spec<br>• 1398: new classifications in content descriptor requested by UK DTG<br>• 1399: typo in clause 6.4.5.4<br>• 1420: table K.1 typo<br>• 1450: UHD-1 Phase 2 signalling<br>• 1458: unify wording to sub_table<br>• 1463: add user defined range for data_service_id<br>• 1710: next-generation audio (NGA) signalling<br>• 1713: clarify MSB/LSB for uimsbf fields in definitions<br>• 1714: signalling support for TM-MIS authentication solution in ETSI TS 102 809<br>• 1736: replace BlueBook reference in table 14<br>• 1737: add signalling for TTML subtitles<br>• 1744: update codec references to point to ETSI TS 101 154 instead of/in addition to the codec specs<br>• 1951: reference MPEG for CRC32 (void annex B)<br>• 1953: stale ISO/IEC 13818-1:2007/AMD1 reference<br>• 2065: enhanced modes of CSA3 were never specified<br>• 2066: ambiguous reference to ATIS 0800006<br>• 2096: harmonize descriptor naming for DTS descriptors<br>• 2099: URI linkage type registration solution<br>• 2176: next-generation audio overview<br>• 2206: promote ETSI TS 101 162 to a normative reference<br>• 1008: adaptation_field_data_identifier for TSAP timeline (not)<br>• 1417: S2 satellite delivery descriptor - wording and referencing issues.<br>• 1418: is the S2 delivery system descriptor stand-alone or used together with an S delivery system descriptor?<br>• 1419: remove references to S2 NBC-BS mode<br>• 1479: Incomplete title for table 26: stream_content and component_type<br>• 1570: new section to clarify of bit order and transmission order<br>• 1711: inconsistent table lines in table 26: stream_content and component_type<br>• 1718: annex N changelog corrections for v1.15.1<br>• 2053: ambiguous wording implying a descriptor ordering in clause 6.2.39 Stream Identifier Descriptor<br>• 2168: typo in heading of clause 6.4.16.0<br>• 2170: clause 6.2.1 Adaptation field data descriptor: add reference to annex D of ETSI TS 101 154<br>• 2183: Use of the supplementary audio descriptor with AC-4<br>• 2279: typo in note 4 to table 89 (service type) |
| October 2021 | 1.17.1 | Implemented Change Requests:<br>• 2803: clause 5.1.3, table 2: table_id 0x4C assigned more than once<br>• Addition of 8k HEVC signalling<br>• Addition of support for dynamic HDR metadata |
| March 2023 | 1.18.1 | • Fixing typos introduced in v1.17.1<br>• Adding SI signalling for VVC and AVS3 codecs.<br>• Added VVC subpicture descriptor<br>• Adding DVB-S2Xv2 (Beamforming) SI signalling and Satellite access tables.<br>• TM-NIP bootstrap linkage descriptor type |
| June 2024 | 1.19.1 | • Added Service Prominence Descriptor<br>• Added new Satellite_position_v3 SAT table<br>• Backwards compatible revision of DVB S2Xv2 satellite delivery descriptor. |

# History

| Document history | | |
|---|---|---|
| Edition 1 | October 1995 | Publication as ETSI ETS 300 468 |
| Edition 2 | January 1997 | Publication as ETSI ETS 300 468 |
| V1.3.1 | February 1998 | Publication |
| V1.4.1 | November 2000 | Publication |
| V1.5.1 | May 2003 | Publication |
| V1.6.1 | November 2004 | Publication |
| V1.7.1 | May 2006 | Publication |
| V1.8.1 | July 2008 | Publication |
| V1.9.1 | March 2009 | Publication |
| V1.10.1 | November 2009 | Publication |
| V1.11.1 | April 2010 | Publication |
| V1.12.1 | October 2011 | Publication |
| V1.13.1 | August 2012 | Publication |
| V1.14.1 | May 2014 | Publication |
| V1.15.1 | March 2016 | Publication |
| V1.16.1 | August 2019 | Publication |
| V1.17.1 | October 2022 | Publication |
| V1.18.1 | December 2023 | Publication |
| V1.19.0 | November 2024 | EN Approval Procedure          AP 20250216: 2024-11-18 to 2025-02-17 |