ETSI EN 300 743 V1.6.1 (2018-10)

EUROPEAN STANDARD

Digital Video Broadcasting (DVB);
Subtitling systems

EBU
OPERATING EUROVISION

Reference

REN/JTC-DVB-378

Keywords

broadcasting, digital, DVB, TV, video

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is 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 Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, 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.

Foreword

This 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).

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 Digital Video Broadcasting Project (DVB) is an industry-led consortium of broadcasters, manufacturers, network operators, software developers, regulatory bodies, content owners and others committed to designing global standards for the delivery of digital television and data services. DVB fosters market driven solutions that meet the needs and economic circumstances of broadcast industry stakeholders and consumers. DVB standards 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 to provide global standardization, interoperability and future proof specifications.

National transposition dates	
Date of adoption of this EN:	22 October 2018
Date of latest announcement of this EN (doa):	31 January 2019
Date of latest publication of new National Standard or endorsement of this EN (dop/e):	31 July 2019
Date of withdrawal of any conflicting National Standard (dow):	31 July 2019

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 method by which subtitles, logos and other graphical elements may be coded and carried in DVB bitstreams. The system applies Colour Look-Up Tables (CLUTs) to define the colours of the graphical elements. The transport of the coded graphical elements is based on the MPEG-2 Transport Stream described in ISO/IEC 13818-1 [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 300 468: "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems".

[3] Recommendation ITU-R BT.601: "Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios".

[4] Recommendation ITU-R BT.656-4: "Interface for digital component video signals in 525-line and 625-line television systems operating at the 4:2:2 level of Recommendation ITU-R BT.601 (Part A)".

[5] ETSI EN 300 743 (V1.2.1): "Digital Video Broadcasting (DVB); Subtitling systems".

[6] ETSI EN 300 743 (V1.3.1): "Digital Video Broadcasting (DVB); Subtitling systems".

[7] ETSI EN 300 743 (V1.4.1): "Digital Video Broadcasting (DVB); Subtitling systems".

[8] ETSI EN 300 743 (V1.5.1): "Digital Video Broadcasting (DVB); Subtitling systems".

[9] ETSI TS 101 154: "Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream".

[10] Recommendation ITU-R BT.709: "Parameter values for the HDTV standards for production and international programme exchange".

[11] Recommendation ITU-R BT.2020-2: "Parameter values for ultra-high definition television systems for production and international programme exchange".

[12] Recommendation ITU-R BT.2100-1: "Image parameter values for high dynamic range television for use in production and international programme exchange".

[13] Recommendation ITU-R BT.1886: "Reference electro-optical transfer function for flat panel displays used in HDTV studio production".

[14] IETF RFC 1950: "ZLIB Compressed Data Format Specification", (Version 3.3).

[15] IETF RFC 1951: "DEFLATE Compressed Data Format Specification", (Version 1.3).

[16] ISO/IEC 15948: "Information technology, Computer graphics and image processing, Portable Network Graphics (PNG): Functional specification".

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

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: 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.

Not applicable.

3 Definitions and abbreviations

3.1 Definitions

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

ancillary page: means of conveying subtitle elements that may be shared by multiple **subtitle services** within a **subtitle stream**, used for example to carry logos or character glyphs

Colour Look-Up Table (CLUT): look-up table applied in each region for translating the objects' pseudo-colours into the correct colours to be displayed

CLUT family: family of CLUTs sharing the same CLUT_id, which consists of three CLUTs for rendering in Recommendation ITU-R BT.601 [3] colour space, each of which is pre-populated with a **default CLUT**; - one with 4 entries, one with 16 entries, and one with 256 entries; and optionally one or more alternative CLUTs (provided in the ACS) for rendering in colour and/or dynamic range systems other than Recommendation ITU-R BT.601 [3]

composition page: means of conveying subtitle elements for one specific **subtitle service**

default CLUT: CLUT populated with a set of preset colour entries that provide a useful range of colours within the limit of the maximum number of entries in the respective CLUT

display definition: definition of the video image display resolution for which a subtitle stream has been prepared

display set: set of **subtitle segments** of a specific **subtitle service** to which the same **PTS** value is associated

epoch: period of time for which the decoder maintains an invariant memory layout, in the form of the defined **page compositions**

next_bits(n): function that provides the next 'n' bits in the bitstream, without advancing the bitstream pointer, which permits the comparison of those bit values with another sequence of bit values of the same length

object: graphical unit, identified by its own object_id, that can be positioned within a **region**; examples of an object include a character glyph, a logo, a map, etc.

Packet IDentifier (PID): Transport packet identifier, as defined in ISO/IEC 13818-1 [1].

page: set of subtitles for a **subtitle service** during a certain period, consisting of one or more **page instances** whereby each page update or refresh will result in a new page instance. A page contains a number of **regions**, and in each region there can be a number of **objects**

page composition: composition (use and positioning) of **regions** that may be displayed within the **page**, whereby only one page composition is active for displaying at any one time, and changes can occur at any new **page instance**, for example some regions might not be displayed yet, or some regions might no longer be displayed

page instance: period of time, typically initiated with the **PTS** of a **display set,** during which that **page** does not change i.e. there is no change to the **page composition**, to any **region composition**, to any **object** within a **region** or any applicable **CLUT**

PES packet: See ISO/IEC 13818-1 [1].

pixel-data: string of data bytes that contains, in coded form, the representation of a graphical object

Presentation Time Stamp (PTS): See ISO/IEC 13818-1 [1].

region: rectangular area on the **page** in which **objects** can be positioned

region composition: composition (use and positioning) of **objects** within a **region**

reserved: When used in a clause defining the coded bit stream, this field indicates that the value may be used for extensions in the future. Unless specified otherwise within the present document, all "reserved" bits are expected to be set to "1".

reserved_zero_future_use: When used in clauses defining the coded bit stream, this field indicates that the value may be used in future revisions for ETSI-defined extensions. All "reserved_zero_future_use" bits are expected to be set to "0".

subtitle element: subtitle data used within a **page composition** and contained within a **subtitle segment**, for example r**egions**, **region composition**s, **CLUT** definitions and **object** data

subtitle segment: basic syntactical element of a **subtitle stream**

subtitle service: service, displayed as a series of one or more **pages**, that provides subtitling for a program for a certain purpose and to satisfy a single communication requirement, such as subtitles in a specific language for one program or subtitles for the hard of hearing

subtitle stream: stream containing one or more **subtitle services** and consisting of **subtitling segment**s carried in **transport packet**s identified by the same **PID**

transport packet: See ISO/IEC 13818-1 [1].

transport stream: stream of **transport packet**s carrying one or more MPEG programs, as defined in ISO/IEC 13818-1 [1].

3.2 Abbreviations

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

3DTV plano-stereoscopic Three-Dimensional TeleVision
ACS Alternative CLUT Segment
B Blue value of colour representation in default CLUT
bslbf bit string, left bit first
Cb Chrominance value representing B-Y colour difference signal

NOTE: As defined in Recommendation ITU-R BT.601 [3], clause 7.2.3.

CLUT Colour Look-Up Table
CLUT_id CLUT identifier
Cr Chrominance value representing R-Y colour difference signal

NOTE: As defined in Recommendation ITU-R BT.601 [3], clause 7.2.3.

DDS Display Definition Segment
DSS Disparity Signalling Segment
DTV Digital TeleVision
DVB Digital Video Broadcasting

EDS End of Display Set Segment

NOTE: As defined in ETSI EN 300 468 [2].

G Green value of colour representation in default CLUT
GOP Group Of Pictures
HDR High Dynamic Range
HDTV High Definition TeleVision
HLG Hybrid Log-Gamma
IRD Integrated Receiver Decoder
MPEG Moving Pictures Experts Group

NOTE: WG11 in SC 29 of JTC1 of ISO/IEC.

PCR Programme Clock Reference

NOTE: As defined in ISO/IEC 13818-1 [1].

PCS Page Composition Segment
PES Packetized Elementary Stream

NOTE: As defined in ISO/IEC 13818-1 [1].

PID Packet IDentifier

NOTE: As defined in ISO/IEC 13818-1 [1].

PMT Program Map Table

NOTE: As defined in ISO/IEC 13818-1 [1].

PNG Portable Network Graphics

NOTE: As defined in ISO/IEC 15948 [16].

PQ Perceptual Quantization
PSI Program Specific Information

NOTE: As defined in ISO/IEC 13818-1 [1].

PTS Presentation Time Stamp

NOTE: As defined in ISO/IEC 13818-1 [1].

R Red value of colour representation in default CLUT
RCS Region Composition Segment
ROM Read-Only Memory
SDR Standard Dynamic Range

NOTE: As defined in ETSI EN 300 468 [2].

SDTV Standard Definition TeleVision
SI Service Information
STC System Time Clock

NOTE: As defined in ISO/IEC 13818-1 [1].

T Transparency value
TS Transport Stream

NOTE: As defined in ISO/IEC 13818-1 [1].

UHDTV Ultra-High Definition TeleVision
uimsbf unsigned integer, most significant bit first
tcimsbf two's complement integer, msb (sign) bit first

Y luminance value

NOTE: As defined in Recommendation ITU-R BT.601 [3], clause 7.2.3.

CDS	CLUT Definition Segment
ODS	Object Data Segment
RGB	Red Green Blue
CRC	Cyclic Redundancy Check
AV	Audio Video
PLTE	Palette Table
IDAT	Image Data
IHDR	Image Header

4 Introduction to the DVB subtitling system (informative)

4.1 General

The present clause provides an informative introduction to the DVB subtitling system.

Clause 4.2 first provides an account of the evolution of the present document in relation to the continual enhancement of video formats used by DVB services.

Clause 4.3 introduces the basic concepts and terminology for DVB subtitling.

Clause 4.4 describes the composition of the DVB subtitling data structure.

Clause 4.5 describes the DVB subtitling segment coding method.

Clause 4.6 describes the method of transport of DVB subtitling content.

Clause 4.7 describes the subtitling data hierarchy.

Clause 4.8 describes the subtitling temporal hierarchy and terminology.

The normative specification of the subtitling system is contained in clauses 5 to 10.

4.2 Subtitling system evolution and service compatibility

4.2.1 Introduction

The present document has been revised several times in order to introduce new features and maintain its applicability as new types of DVB service emerged, namely HDTV, 3DTV, and most recently UHDTV, all of which are specified in ETSI TS 101 154 [9] as regards codec usage, and ETSI EN 300 468 [2] as regards signalling. Maintenance revisions have been made in addition to these. The remainder of the present clause summarizes the history of the present document in relation to its revisions, whereas clause 4.2.6 provides a summary of subtitle service compatibility issues resulting from the multiple versions of the present document.

4.2.2 V1.1.1 and V1.2.1

The first edition of the present document [17], published in 1997, specified the subtitling system only for SDTV services, as defined in ETSI TS 110 154 [9] and ETSI EN 300 468 [2].

ETSI EN 300 743 [5] (V1.2.1) of the present document was a general maintenance revision.

4.2.3 V1.3.1

ETSI EN 300 743 [6] (V1.3.1) of the present document added support for subtitles for HDTV services, as defined in ETSI TS 101 154 [9] and ETSI EN 300 468 [2]. In V1.3.1 a new optional segment was specified, namely the **display definition segment** (DDS). The DDS explicitly defines the display resolution for which that stream has been created, i.e. it allows subtitles with display resolutions other than that for SDTV to be provided, and optionally allows subtitles to be positioned within a window that constitutes only a part of the full display resolution.

The DDS is not needed with subtitle streams associated with SDTV services, thus they can be encoded in accordance with ETSI EN 300 743 (V1.2.1) [5]. Such streams will nevertheless be decodable by decoders compliant with any later versions of the present document.

Subtitles are encoded using Recommendation ITU-R BT.601 [3] colorimetry, i.e. the same as that used in video components of SDTV services. HDTV systems use Recommendation ITU-R BT.709 [10] colorimtery, but that distinction was not taken into account when the present document was revised to include HDTV-resolution subtitles in ETSI EN 300 743 (V1.3.1) [6]. Due to the small differences between these two systems, in practice it does not matter which one is ultimately used to render the subtitles.

4.2.4 V1.4.1 and V1.5.1

Version 1.4.1 of the present document added support for subtitles for 3DTV services, as defined in ETSI TS 101 154 [9] and ETSI EN 300 468 [2]. In ETSI EN 300 743 (V1.4.1) [7] a new optional subtitling segment was specified, namely the **disparity signalling segment** (DSS). The DSS enables a region or part of a region to be attributed with a disparity value, to facilitate the optimal rendering of subtitles over 3DTV content.

ETSI EN 300 743 (V1.5.1) [8] of the present document was a general maintenance revision.

4.2.5 V1.6.1

Version 1.6.1 of the present document adds explicit support of subtitling for UHDTV services, as defined in ETSI TS 101 154 [9] and ETSI EN 300 468 [2].

The latest revision of the present document, V1.6.1, introduces technical extensions specifically for progressive-scan subtitle object coding and the capability to provide the subtitle CLUT in other colour systems in addition to Recommendation ITU-R BT.601 [3]. These extensions are partitioned clearly by the definition of a new *subtitling_type* to be used in subtitle services that make use of any of these new features, so that no changes for existing implementations according to V1.5.1 or earlier versions of the present document are implied.

Video content for UHDTV services, as defined in ETSI TS 110 154 [9], uses Recommendation ITU-R BT.2020-2 [11] colorimetry. Since that colour system is vastly enhanced compared to Recommendation ITU-R BT.709 [10] and Recommendation ITU-R BT.601 [3], the capability was introduced in V1.6.1 of the present document to enable the subtitle service to provide the CLUT for rendering in Recommendation ITU-R BT.2020-2 [11] / Recommendation ITU-R BT.2100-1 [12] colour volume, using the **alternative CLUT segment** (ACS), in addition to the legacy CLUT(s) defined by the default CLUTs and the CLUT definition segment (CDS). This allows IRDs that support graphics rendering in Recommendation ITU-R BT.2020-2 [11] and, if applicable, HDR to render directly both video and bitmap subtitles without conversion of colour and dynamic range.

Version 1.6.1 of the present document also adds support of subtitles in progressive-scan format, whereby the subtitle objects are in a format that can be converted conveniently from a suitably coded PNG [16] file. Such subtitle objects are not compatible with IRDs that were designed to be compatible with version 1.5.1 of the present document or earlier.

The DDS is also included in subtitle streams intended for UHDTV services, whereby subtitle graphics rendering is constrained to HDTV resolution. Where the display window feature of the DDS is not used, the UHDTV IRD upscales the subtitles spatially before rendering them on a UHDTV resolution display.

Subtitle streams associated with SDTV or HDTV services and intended to be decoded by decoders designed to ETSI EN 300 743 (V1.5.1) [8] or an earlier version naturally make use of neither the alternative CLUT segment (ACS), nor of the new object coding method for progressively coded subtitle bitmaps.

4.2.6 General subtitle service compatibility issues

While there is no explicit signalling in subtitling streams of the version of the present document with which the stream conforms, an implicit signalling of compatibility is provided via the subtitling_descriptor carried in the PMT of the associated service. The appropriate setting of the subtitling_type field of the subtitling descriptor in the PMT is thus important for subtitle decoder compatibility. IRDs are expected to ignore subtitle services signalled with a subtitling_type that they do not support.

Table 1 provides an overview of the evolution of the present document in order to state indicatively the compatibility of different versions of the present document with the various forms of DVB service.

For subtitle decoders, the principle of "backwards compatibility" applies, i.e. decoders that are implemented to support a particular version of the present document are expected to support all features of all previous versions of the specification that are relevant for that decoder. The DSS is relevant only for IRDs that support 3DTV services.

For subtitle services, the principle of "decoder compatibility" applies, i.e. the service can adopt any feature from a particular version of the present document in line with the subtitling type value used. Clause 6.3 specifies this aspect normatively.

Table 1: Evolution of the subtitling specification

ETSI EN 300 743 version	Services supported				Major new features
	SDTV	HDTV	3DTV	UHDTV	
V1.1.1, V1.2.1	X				Not applicable
V1.3.1	X	X		X (see note 1)	Signalling of intended resolution using the Display Definition Segment (DDS)
V1.4.1, V1.5.1	X	X	X	X (see note 1)	Signalling of disparity for subtitles using the Disparity Signalling Segment (DSS)
V1.6.1	X (see note 2)	X (see note 2)	X (see note 2)	X	Progressive encoding (using new object coding type in the Object Data Segment (ODS)); CLUT provision for other colour spaces, using the Alternative CLUT Segment (ACS)
NOTE 1: The subtitle service defines the available subtitle colours according to Recommendation ITU-R BT.601 [3] only, using the default CLUTs and/or the CLUT definition segment (CDS). NOTE 2: If the subtitle service uses ODS object coding type = '2' then decoders compliant with V1.5.1 or earlier of the present document will not be able to decode the subtitles.					

The two new features introduced in version 1.6.1 of the present document could, in principle, also be used with non-UHDTV service types. These features are the alternative CLUT segment and progressive-scan bitmap objects. A DVB service is not obliged to provide subtitles with capabilities according to the most advanced version of the present document, nor necessarily according to the indicative service compatibility as shown in table 1. For example, a UHDTV service could include subtitle streams that are conformant with V1.3.1 of the present document, if the service provider chooses to do so, bearing in mind that there might be unpredictable results with the positioning of such subtitles on the screen with some UHDTV IRDs.

Conversely, if a service provider wishes to deploy progressively-coded subtitles (with ODS object coding type = '2'), the subtitle stream will be conformant with V1.6.1 of the present document, and is signalled as such even if the service is not a UHDTV service.

Subtitle service signalling is specified normatively in clause 6.3.

4.3 Basic concepts and terminology

The DVB subtitling system provides a syntax for delivering **subtitle streams**. A subtitle stream conveys one or more **subtitle services**; each service containing the textual and/or graphical information needed to provide subtitles or glyphs for a particular purpose. Separate subtitle services may be used, for example, to convey subtitles in several languages.

Different subtitle services can also be supplied to address different display characteristics, for instance SDTV resolution subtitles can be provided for 4:3 and 16:9 aspect ratio displays, using subtitling_type signalling to distinguish between the subtitle streams. Different subtitle services might address special needs, for instance specifically for viewers with impaired hearing; these may include graphical representations of sounds.

Each subtitle service displays its information in a sequence of so-called **pages** that are intended to be overlayed on the associated video image. A subtitle page contains one or more **regions**, each region being a rectangular area with a specified set of attributes. These attributes include a region identifier, the horizontal and vertical resolution, pixel depth and background colour. A region is used as the background structure into which graphical **objects** are placed. An object may represent a character, a word, a line of text or an entire sentence; it might also define a logo or icon.

Figure 1 depicts an example subtitling page that consists of two regions. Here, region 1 is used to display a static logo in the top-right corner of the screen, while region 2 is used to display multiple subtitle fragments. First the text "Shall we?" is displayed in the region; subsequently this text is removed and the new text "Said the fly on the mirror" is displayed within the same region.

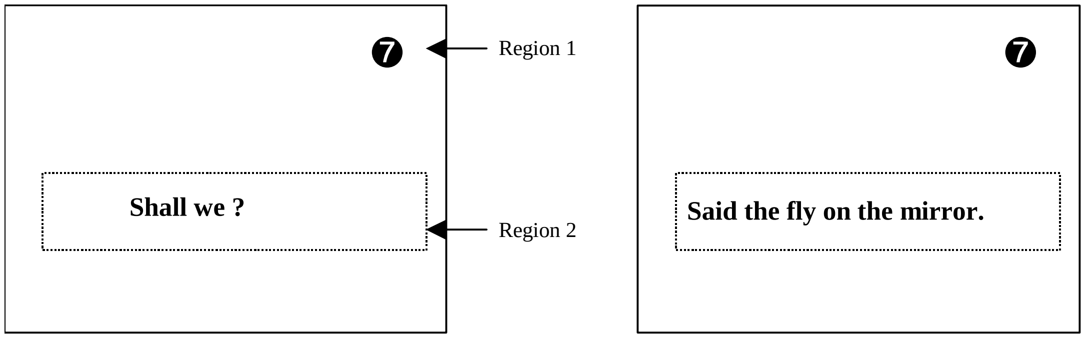


Figure 1: Two regions overlayed on top of video

A sequence of one or more subtitling page instances that re-use certain initial properties is referred to as an **epoch**. The page composition and the region composition may change within an epoch - for example objects and regions may be added or removed. The concept of an epoch is analogous to that of an MPEG video sequence; no decoder state is preserved from one epoch to the next.

Each page instance is a completed screen of graphics. Consecutive page instances may differ little (e.g. by a single word when stenographic subtitling is being used) or may be completely different. The basic "building block" of a DVB subtitle stream is the subtitling **segment**. Several segment types are defined for the carriage of the various types of subtitling data.

The set of segments constituting a single page instance is referred to as a **display set**.

4.4 Subtitle stream composition

The following segment types are defined in clause 7.2, in order to fulfill the various functions around the provision of subtitle services. The order of their listing here matches their ordering, when present, in a display set:

- **display definition segment**; a subtitle service may be intended or have been prepared for display resolutions other than SDTV (i.e. other than 720 by 576 pixels -e.g. for HDTV). The optional display definition segment explicitly defines the display resolution for which that service has been created;
- **page composition segment**; the decoding of a subtitle service will typically result in the display of subsequent pages, each consisting of one or more regions; the page composition segment carries information on the page composition, such as the list of included regions, the spatial position of each region, some time-out information for the page and the state of the page. The addition or removal of objects within a region does not necessarily change the page composition. Furthermore regions may be declared but not used. It is possible to use more than one region at the same time;
- **region composition segment**; in each region typically one or more objects are positioned, while using one specific CLUT family, identified by a CLUT_id; the region composition segment carries information on the region composition and on region attributes, such as the horizontal and vertical resolution, the background colour, the pixel depth of the region, which CLUT family is used and a list of included objects with their position within the region;
- **disparity signalling segment**; this segment type supports the subtitling of plano-stereoscopic 3DTV content by allowing disparity values to be ascribed to a region or to part of a region. This segment type is not used with non-3DTV services;

- **CLUT definition segment**; this segment type contains information on a specific CLUT family, identified by a CLUT_id, namely any replacements of the colours defined in the default CLUT(s) used for for rendering objects in the Recommendation ITU-R BT.601 [3] colour space. Up to three CLUTs can be defined for the CLUT family using the CDS, for use by decoders with different rendering capabilities;
- **alternative CLUT segment**; this segment type contains information on a specific CLUT family, identified by the CLUT_id, such as the colours used for CLUT entries as an alternative to the CDS. This segment type is used to convey the subtitling CLUT used for colour systems other than the Recommendation ITU-R BT.601 [3] colour space - for example Recommendation ITU-R BT.2100-1 [12] with HDR;
- **object data segment**; the object data segment carries information on a specific subtitle object. Objects that occur more than once within a region need only be transmitted once, and then positioned multiple times within the region. Objects used in more than one subtitle service need only be transmitted once. There are three types of object:
    - A graphical object in interlaced format that contains run-length encoded bitmap colours.
    - A graphical object in progressive format, whose sequence of filtered scanlines is compressed using *zlib* [14], which in turn applies *DEFLATE* compression [15], as defined in the PNG [16] graphics format.
    - A text object that carries a string of character codes. Usage of the text object is not defined in the present document.
- **end of display set segment**; the end of display set segment contains no internal information, but is used to signal explicitly that no more segments need to be received before the decoding of the current display set can commence;

Segment types that are not recognized or supported are expected to be ignored.

The display sets of a subtitle service are delivered in their correct presentation order, and the PTSs of subsequent display sets differs by at least one video frame period.

4.5 Subtitle segment coding

To provide efficient use of display memory in the decoder the DVB subtitling system uses region based graphics with indexed pixel colours that are contained in a Colour Look-Up Table (**CLUT**). The original version of the present document was published at a time when IRD graphics capabilities were relatively limited. For rendering in Recommendation ITU-R BT.601 [3] colour space, three CLUTs are defined in order to take into account such potential limitations in the decoder. Pixel depths of 2, 4 and 8 bits are supported allowing up to 4, 16 or 256 different pixel codes to be used. Not all decoders supported a CLUT with 256 entries; some provided sixteen or even only four entries. A palette of four colours might be enough for graphics that are basically monochrome, like very simple subtitles, while a palette of sixteen colours allows for cartoon-like coloured objects or coloured subtitles with antialiased edges.

Each region is associated with a single **CLUT family** to define the colour and transparency for each of the pixel codes. In most cases, one CLUT family is sufficient to present correctly the colours of all objects in a region, but if it is not enough, the objects can be split horizontally into smaller objects across separate vertically adjacent regions with one CLUT family each.

The use of CLUTs allows colour schemes to be dynamic. The colours that correspond to the entries within the region can be redefined at any suitable time, for instance in case of a CLUT family with four entries from a black-grey-white scheme to a blue-grey-yellow scheme. Furthermore, a graphical unit can be divided into several regions each using a different CLUT family, i.e. a different colour scheme may be applied in each of the regions. At the discretion of the encoder, objects designed for displays supporting 16 or 256 colours can be decoded into displays supporting fewer colours. A quantization algorithm is defined to ensure that the result of this process can be predicted by the originator. Use of this feature allows a single data stream to be decoded by a population of decoders with mixed, and possibly evolving, capabilities.

Where the gamut of colours required for part of a graphical object is suitably limited, that part can be coded using a smaller number of bits per pixel and a map table. For example, an 8-bit per pixel graphical object may contain areas coded as 4 or 2-bits per pixel each preceded by a map table to map the 16 or 4 colours used onto the 256 colour set of the region. Similarly, a 4-bit per pixel object may contain areas coded as 2-bits per pixel.

Default CLUTs are defined in clause 10. These contain a full complement of colours for each of the 2-bit, 4-bit and 8-bit CLUTs.

Colour definitions can be coded using either 16 or 32 bits per CLUT entry. This provides a trade-off between colour accuracy and transmission bandwidth. Only those CLUT values to be used and that are not contained in the default CLUT need be transmitted, by using the CLUT definition segment (CDS).

Version V1.6.1 of the present document extended the definition of CLUT_family by adding the possibility to provide CLUTs for rendering in colour systems other than Recommendation ITU-R BT.601 [3], by using the alternative CLUT segment (ACS). The ACS allows up to a maximum of 256 CLUT entries to be defined. The default 8-bit CLUT is not applicable when using the ACS, hence the full CLUT of colours used is always provided. A CDS is always provided along with the ACS, when used.

Subtitles that apply a contrasting opaque background to the subtitle text provide optimal readability of subtitles.

Two alternative methods to encode subtitle bitmaps are included in the object data segment. The first is the original method specified in clauses 7.2.5.1 and 7.2.5.2. Here, pixel data within objects is comprised of two interlaced fields, or a single repeated field, and compressed using the run-length coding method that is also defined in clauses 7.2.5.1 and 7.2.5.2. The second method, introduced in V1.6.1 of the present document, uses progressively-coded pixel data objects whose scanlines are compressed using *zlib* [14], which in turn applies *DEFLATE* compression [15], as defined in the PNG [16] graphics format.

Additional functionality is provided to allow more efficient operation where there are private agreements between the data provider and the manufacturer of the decoder:

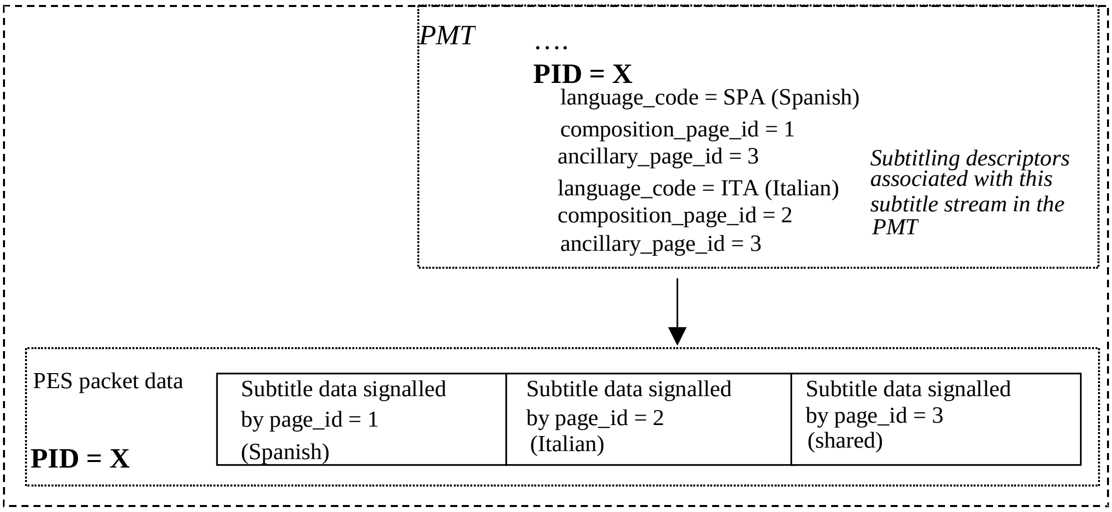
- objects resident in ROM in the decoder can be referenced;
- character codes, or strings of character codes, can be used instead of objects with the graphical representation of the character(s). This requires the decoder to be able to generate glyphs for these codes.

The private agreements required to enable these features are beyond the scope of the present document.

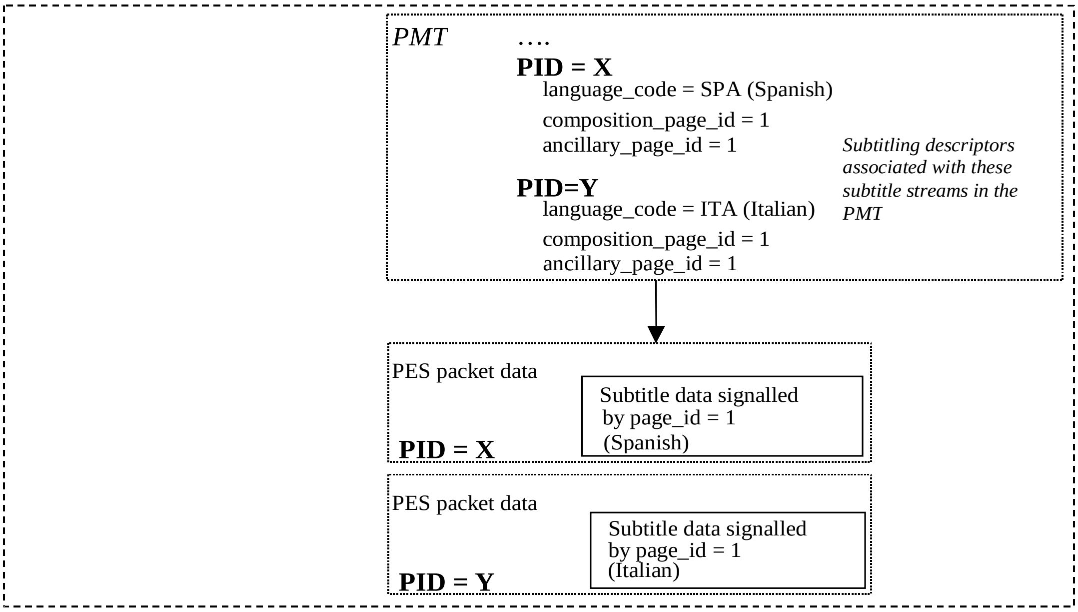
4.6 Subtitle stream transport

A DVB subtitle stream is carried within **PES packets** in the **MPEG-2 Transport Stream** (TS) according to ISO/IEC 13818-1 [1], as specified in clauses 6.2 and 6.3.

A single subtitle stream can carry several different subtitle services, but all the subtitling data required for a subtitle service is carried within a single subtitle stream. The different subtitle services can be subtitles in different languages for a common program. Alternatively, they could in principle be for different programs, provided that the programs share a common **PCR**. In the case of multiple subtitle services in one stream, the pages of each subtitle service are identified by the same page-id value. The subtitling system allows sharing of subtitling data between services within the same subtitle stream. However, the recommended method is to convey the distinct services in different streams on separate **PID**s. In either case the appropriate PID, language and page-ids will be signalled in the Program Map Table (**PMT)** for the television service of interest (language and page-id in the subtitling descriptor defined in DVB-SI as defined in ETSI EN 300 468 [2]). These two approaches are illustrated in figure 2.



a: Example of use of different page_ids to distinguish between different subtitle languages for the same service (shown with a shared ancillary page) -- non-recommended method



b: Example of use of PIDs to distinguish between different subtitle languages for the same service (shown with no ancillary page) - recommended method

Figure 2: Example of two ways of conveying dual language subtitles (one using shared data)

Subtitle streams intended for HDTV, 3DTV or UHDTV services and which include a display definition segment are distinguished from those which are intended for SDTV services and that have been coded in accordance with ETSI EN 300 743 (V1.2.1) [5], by the use of HDTV-specific, 3DTV-specific or UHDTV-specific **subtitling_type** values in the subtitling descriptor signalled in the PMT for that service. The **subtitling_type** value is set to the same value as the **component_type** value of a DVB component descriptor as defined in ETSI EN 300 468 [2] when the stream_content field of that descriptor is equal to '0x3'. This provides a means whereby only subtitle decoders compliant with ETSI EN 300 743 (V1.3.1) [6] or later are expected to be presented with streams that include display definition segments.

For each subtitle service a **subtitling_descriptor** as defined in ETSI EN 300 468 [2] signals the page id values of the segments needed to decode that subtitle service. The subtitling descriptor is included in the PMT of the program and is associated with the PID that conveys the subtitle stream. In the subtitling descriptor the page id of segments with data specific to that service is referred to as the **composition page id**, while the page id of segments with shared data is referred to as the **ancillary page id**. For example, the ancillary page id might signal segments carrying a logo that is common to subtitles in several different languages.

Clause 6.3 specifies completely the requirements for signalling of subtitle services depending on which features are included, their applicability to the different kinds of DVB service, and implications for the IRD.

The **PTS** in the PES packet header provides presentation timing information for the subtitling objects, and is associated with the subtitle data in all segments carried in that PES packet. The PTS defines the time at which the associated decoded segments are presented. This can include removal of subtitles, for example when an entire region is removed or when all objects in a region are removed.

When objects are to be added, the decoder receives region composition updates and the data for the new objects, and displays the updated page at the time indicated by the new PTS. At the page update only page differences need be provided. To improve random access to a DVB subtitle service, a page refresh is also possible. At page refresh all the subtitling data needed to display a page is provided. Each page update or refresh will result in a new page instance. A page ceases to exist after the time-out of the page, or when a new page is defined.

4.7 Subtitling data hierarchy

In summary, the subtitling data hierarchy is:

- Transport Stream (TS);
- transport packets with the same PID;
- PES packets, with PTSs providing timing information;
- subtitle service;
- segments signalled by the composition page id and optionally the ancillary page id;
- where appropriate, a display definition segment;
- subtitle data, containing information on page composition, region composition, disparity signalling (if applicable), CLUTs, objects and end of display set.

4.8 Temporal hierarchy and terminology

At the segment level in the data hierarchy there is also a temporal hierarchy. The highest level is the epoch; in an epoch the page composition and the region composition may change - for example objects and regions may be added or removed. The concept of an epoch is analogous to that of an MPEG video sequence. No decoder state is preserved from one epoch to the next.

An epoch is a sequence of one or more page instances. Each page instance is a completed screen of graphics. Consecutive page instances may differ little (e.g. by a single word when stenographic subtitling is being used) or may be completely different. The set of segments needed to decode a new page instance is called a display set.

Within a display set the sequence of segments (when present) is:

- display definition segment;
- page composition;
- region composition;
- disparity signalling (if applicable);
- CLUT definition(s), using the CDS;
- alternative CLUT definition(s), using the ACS;
- object data;
- end of display set segment.

5 Subtitle decoder model and subtitle stream provision

5.0 Introduction

The subtitle decoder model is an abstraction of the processing required for the decoding of a subtitle service within a subtitle stream. The main purpose of this model is to define requirements for compliant subtitling streams. Figure 3 shows the prototypical model of a subtitling decoder.

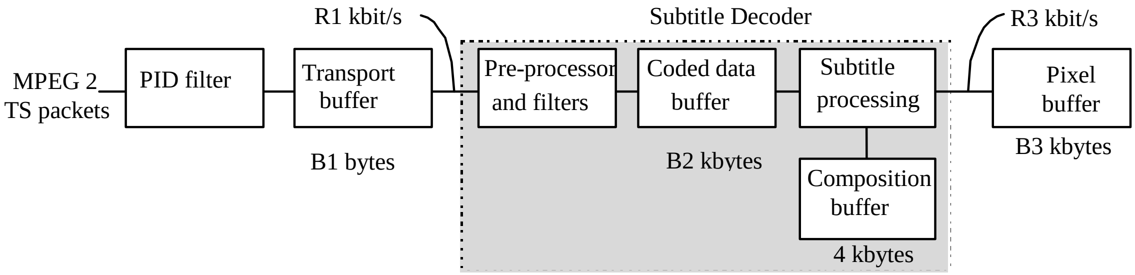


Figure 3: Subtitle decoder model

The input to the subtitle decoder model is an MPEG-2 Transport Stream (TS). After a selection process based on PID value, complete MPEG-2 Transport Stream packets containing the subtitle stream enter a transport buffer with a size of B1 bytes. When there is data in the transport buffer, data is removed from this buffer at a rate of R1 kbit/s. When no data is present, this data rate equals zero. For legacy decoders designed in accordance with ETSI EN 300 743 (V1.2.1) [5] the transport buffer has a size of 512 bytes and the outflow rate is 192 kbits/s. For decoders capable of dealing with streams which include a display definition segment the transport buffer has a size of 1 024 bytes and the outflow rate is 400 kbits/s.

The transport packets from the transport buffer are processed by stripping off the headers of the transport packets and of the PES packets. The Presentation Time Stamp (PTS) values are passed on to the next stages of the subtitling processing. In the pre-processor, the segments required for the selected subtitle service are filtered from the subtitle stream. Hence, the output of the pre-processor is a stream of subtitling segments which are filtered based on the page_id values signalled in the subtitling descriptor.

The selected segments enter into a coded data buffer which has a size of B2 kbytes. For legacy decoders designed in accordance with ETSI EN 300 743 (V1.2.1) [5] the coded data buffer has a size of 24 kbytes. For decoders capable of dealing with streams which include a display definition segment the coded data buffer has a size of 100 kbytes. Only complete segments are removed from this buffer by the subtitle decoder. The removal and decoding of the segments is instantaneous (i.e. it takes zero time). If a segment produces pixel data, the subtitle decoder stops removing segments from the coded data buffer until all pixels have been transferred to the pixel buffer. The pixel data of objects that are used more than once, is transferred separately for each use.

The data rate for the transport of pixel data into the pixel buffer is R3 kbit/s and the pixel buffer size B3 kbytes. For legacy decoders designed in accordance with ETSI EN 300 743 (V1.2.1) [5] the data rate of pixel data into the pixel buffer is 512 kbits/s and the pixel buffer size 80 kbytes. For decoders capable of dealing with streams which include a display definition segment the data rate of pixel data into the pixel buffer is 2 Mb/s and the pixel buffer is 320 kbytes.

The data needed for the composition of the subtitles, such as the page composition, the region composition and the CLUTs are stored in the composition buffer, which has a size of 4 kbytes.

5.1 General principles

5.1.0 The subtitle epoch

The requirements for memory usage in the subtitle decoder model depends on the resolution and colour depth of the applied regions in the page. A complete description of the memory usage of the decoder, comprising the pixel buffer and the composition buffer, shall be delivered at the start of each epoch. Hence, epoch boundaries provide guaranteed service acquisition points. Epoch boundaries are signalled by page composition segments with a page state of type "mode change". An epoch is terminated when the start of a new epoch is signalled.

When a PCS with page state of type "mode change" is received by a decoder, i.e. at the start of an epoch, all memory allocations implied by previous segments are discarded, i.e. the decoder state is reset.

All the regions to be used in an epoch shall be introduced by the Region Composition Segments (RCSs) in the display set that accompanies the PCS with page state of "mode change" (i.e. the first display set of the epoch). This requirement allows a decoder to plan all of its pixel buffer allocations before any object data is written to the buffers. Similarly, all of the CLUT entries to be used during the epoch shall be introduced in this first display set. Subsequent segments can modify the values held in the pixel buffer and composition buffer but may not alter the quantity of memory required.

5.1.1 Subtitle service acquisition

The other allowed values of page state are "acquisition point" and "normal case". Each such "acquisition point" and "normal case" results in a new page instance. The "acquisition point" state (like the "mode change" state) indicates that a complete description of the memory use of the decoder is being broadcast. However, the memory usage shall not change. Decoders that have already acquired the service are only required to look for development of the page (e.g. new objects to be displayed). Re-decoding of previously received segments is optional. Decoders trying to acquire the service may treat a page state of "acquisition point" as if it were "mode change".

Use of the page state of "mode change" may require the decoder to remove the graphic display for a short period while the decoder reallocates its memory use. The "acquisition point" state should not cause any disruption of the display. Hence it is expected that the "mode change" state will be used infrequently (e.g. at the start of a programme, or when there are significant changes in the graphic display). The "acquisition point" state will be used every few seconds to enable rapid service acquisition by decoders.

A page state of "normal case" indicates that the set of RCS may not be complete (the set is only required to include the regions whose region data structures - bitmap or CLUT family - are to be modified in this display set). There is no requirement on decoders to attempt service acquisition at a "normal case" display set.

A display set is not required to contain a page composition segment. Within the same page composition for example a region composition may change. If no page composition segment is contained, the page state is not signalled; however, such display set will result in a new page instance equivalent to a "page update".

5.1.2 Presentation Time Stamps (PTS)

Subtitling segments are encapsulated in PES packets, partly because of their capability to carry a Presentation Time Stamp (PTS) for the subtitling data.

Unlike video pictures, subtitles have no innate refresh rate. Therefore all subtitle data are associated with a PTS to control when the decoded subtitle is displayed. Each PES header shall carry a PTS, associated with all the subtitle data contained within that PES packet. Consequently, for any subtitling service there can be at most one display set in each PES packet. However, the PES packet can contain concurrent display sets for a number of different subtitle services, all sharing the same presentation time. It is possible that segments sharing the same PTS have to be split over more than one PES packet (e.g. because of the 64 kbytes limit on PES packet length). In this case more than one PES packet will have the same PTS value. Subtitling segments should not be fragmented across PES boundaries.

In summary, all of the segments of a single display set shall be carried in one (or more) PES packets that have the same PTS value.

For each subtitling service all data of a display set shall be delivered within the constraints defined for the subtitle decoder model, so as to allow practical decoders sufficient headroom to present the decoded data by the time indicated by the PTS.

There may be times when, due for example to slightly late arrival of a complete display set or due to slow rendering in the decoder, the correct time to present a subtitle (i.e. when PTS = local system clock derived from the PCR) has passed. (Late arrival can result from injudicious throttling of the bit-rate assigned to a subtitling stream at some point in the distribution network.) Under such conditions it is almost always better to display a late subtitle than to discard it.

5.1.3 Usage of the display definition segment

If present in the stream, the Display Definition Segment (DDS) defines the display width and height of the TV image into which the associated DVB subtitles are to be rendered (e.g. in the case of HDTV images into 1 920 by 1 080 pixels, into 1 440 by 1 080 pixels, into 1 280 by 720 pixels, etc.). The DDS applies to the subtitle display set being signalled and thus, if present, is transmitted once per display set.

Absence of a DDS in a stream implies that the stream is coded in accordance with ETSI EN 300 743 (V1.2.1) [5]. In this case the decoder shall assume a screen resolution of 720 by 576 pixels.

The DDS includes the option to signal a window within the image display into which DVB subtitles are to be rendered. This facilitates the application to HDTV services of DVB subtitles rendered for SDTV (e.g. for simulcasting SDTV and HDTV). Thus DVB subtitles rendered for a 720 by 576 pixels image can be positioned within the HDTV image in a flexible manner to suit the service provider (e.g. centred horizontally and positioned at the bottom of the HDTV frame).

Subtitles for UHDTV services have the same maximum resolution as subtitles for HDTV services. It is recommended that subtitles for UHDTV services are provided to be rendered into the maximum HDTV resolution, i.e. a display resolution of 1 920 by 1 080 pixels. If no display window is signalled, then the IRD shall upscale subtitles spatially before rendering them on a UHDTV resolution display. If a display window is used, then the subtitles are rendered without upscaling inside a window that has maximum dimensions of the maximum HDTV resolution, within the maximum UHDTV display resolution (3 840 by 2 160 pixels).

It is not recommended to use DVB subtitles generated at SDTV resolution in a UHDTV service.

Annex B provides examples of how the DDS might be used in practice.

5.1.4 Page composition

The Page Composition Segment (PCS) carries a list of zero or more regions. This list defines the set of regions that will be displayed at the time defined by the associated PTS. In the subtitle decoder model, the display instantly switches from any previously existing set of visible regions to the newly defined set.

The PCS may be followed by zero or more Region Composition Segments (RCS). The region list in the PCS may be quite different from the set of RCSs that follow, in particular when some of the regions are initially not displayed.

The PCS provides the list of regions with their spatial positions on the screen or for streams which include a display definition segment their spatial positions relative to the display window. The vertical position of the regions shall be defined such that regions do not share any horizontal scan lines on the screen. A region therefore monopolizes any part of the scan lines that it occupies; no two regions can be presented horizontally next to each other.

5.1.5 Region composition

A complete set of Region Composition Segments (RCS) shall be present in the display set that follows a PCS with page state of "mode change" or "acquisition point" as this is the process that introduces regions and allocates memory for them. Display sets that represent a page update are only required to contain the data to be modified.

Once introduced the memory "foot print" of a region shall remain fixed for the remainder of the epoch. Therefore the following attributes of the region shall not change within an epoch:

- width;
- height;
- depth;
- region_level_of_compatibility;
- CLUT_id.

Other attributes of the region specified in the RCS are the region_fill_flag and the region_n-bit_pixel_code, specifying the background colour of the region. When the region_fill_flag is set the first graphics operation performed on a region should be to colour all pixels in the region with the colour indicated by the region_n-bit_pixel_code. The value of the region_n-bit_pixel_code shall not change in RCS where the region_fill_flag is not set. This allows decoders that have already acquired the subtitling service to ignore the region_n-bit_pixel_code when the region_fill_flag is not set. A decoder in the process of acquiring the service can rely on the region_n-bit_pixel_code being the current region fill colour regardless of the state of region_fill_flag.

There is no requirement for a region to be initialized by filling it with the background colour when the region is introduced at the start of the epoch. This allows the rendering load to be deferred until the region is included in the region list of the PCS, indicating that presentation of the region is required. In the limiting case, the region need never be filled with the background colour. For example, this may occur if the region is completely covered with objects.

Regions can be shared by multiple subtitling services within the same subtitle stream. Objects that share one or more horizontal scan lines on the screen shall be included in the same region.

5.1.6 Points to note

- At the start of an epoch the display set shall include a complete set of RCSs for all the regions that will be used during that epoch. The PCS shall only list the subset of those regions that are presented at the start of the epoch. In the limiting case any PCS may list zero visible regions.
- An RCS shall be present in a display set if the region is to be modified. However, the RCS is not required to be in the PCS region list. This allows regions to be modified while they are not visible.
- RCSs may be present in a display set even if they are not being modified. For example, a broadcaster may choose to broadcast a complete list of RCSs in every display set.
- A decoder shall inspect every RCS in the display set to determine if the region is to be modified, for example, which pixel buffer modifications are required, or where there is a modification to the associated CLUT family. It is sufficient for the decoder to inspect the RCS version number to determine if a region requires modification. There are three possible causes of modification, any or all of which may cause the modification:
    - region fill flag set;
    - CLUT contents modification;
    - a non-zero length object list.

5.2 Buffer memory model

5.2.0 General

A page composition segment with the page state of type "mode change" destroys all previous pixel buffer and composition buffer allocations by erasing the contents of the buffers.

Various processes, as detailed in the following clauses, allocate memory from the pixel and composition buffers. These allocations persist until the next page composition segment with page state of type "mode change".

There is no mechanism to partially re-allocate memory within an epoch. During an epoch, the memory allocation in the pixel buffer remains the same.

5.2.1 Pixel buffer memory

The pixel buffer in the subtitle decoder has a size of 80 kbytes (320 kbytes for decoders capable of dealing with streams which include a display definition segment). The pixel buffer shall never overflow. Up to 75 % is assigned for active display. The remaining capacity is assigned for future display. The subtitle decoder model assumes that all regions used during an epoch are stored in the pixel buffer and defines the following memory allocation requirement for a region in the pixel buffer:

$$region\_bits = region\_width \times region\_height \times region\_depth$$

where region_depth is the region's pixel depth in bits specified in the RCS. A practical implementation of a subtitle decoder may require more memory to store each region. Any such implementation dependent overhead is not taken into account by the subtitle decoder model.

During an epoch, the occupancy of the pixel buffer is the sum of the region_bits of all regions used in that epoch.

5.2.2 Region memory

The pixel buffer memory for a region is allocated at the start of an epoch. This memory allocation is retained until a page composition segment with page state of "mode change" destroys all memory allocations.

5.2.3 Composition buffer memory

The composition buffer contains all information on page composition, region composition and CLUT definition.

The number of bytes defined by the subtitle decoder model for composition buffer memory allocation is given below:

- Page composition except region list 4 bytes
    - per included region 6 bytes
- Region composition except object list 12 bytes
    - per included object 8 bytes
- CLUT definition excluding entries 4 bytes
    - per non full range entry 4 bytes
    - per full range entry 6 bytes

The provision of one or more alternative_CLUT_segments (ACS) in addition to the CLUT_definition_segment (CDS) implies an increased usage of the composition buffer memory. As defined in clause 5.0, and considering that the CDS has a maximum size of 1,5 Kbytes, that the ACS has a maximum size of 1 Kbyte, and that the size of the composition buffer is 4 Kbytes, the provision of two ACSs in addition to the CDS will not cause an over-filling of the composition buffer memory.

5.3 Cumulative display construction

During an epoch the region modifications defined in display sets accumulate in the pixel buffer, but without any impact on the memory allocation for each region.

5.4 Decoder rendering bandwidth model

5.4.0 General

The rendering bandwidth into the pixel buffer is specified as 512 kbit/s (2 Mb/s for decoders capable of dealing with streams which include a display definition segment).

The subtitle decoder model assumes 100 % efficient memory operations. So, when a 10 pixel × 10 pixel object is rendered in a region with a 4-bit pixel depth, 400-bit operations are consumed.

The rendering bandwidth budget comprises all modifications to the pixel buffer. Certain decoder architectures may require a different number of memory operations. For example, certain architectures may require read, modify, write operation on several bytes to modify a single pixel. These implementation dependent issues are beyond the scope of the subtitle decoder model and are to be compensated for by the decoder designer.

5.4.1 Page erasure

A page erasure occurs at a page time-out. Page erasure does not imply any modifications to the pixel buffer. So, page erasure does not impact rendering in the subtitle decoder model.

5.4.2 Region move or change in visibility

Regions can be repositioned by altering the specification of their position in the region list in the PCS. The computational load for doing this may vary greatly depending on the implementation of the graphics system. However, the subtitle decoder model is region based so the model assumes no rendering burden associated with a region move.

Similarly, the visibility of a region can be changed by including it in or excluding it from the PCS region list. As above, the subtitle decoder model assumes that no rendering is associated with modifying the PCS region list.

5.4.3 Region fill

Setting the region fill flag instructs that the region is to be completely re-drawn with the defined fill colour. For example, filling a 128 pixel × 100 pixel 4-bit deep region will consume 51 200 -bit operations, which will take 0,1 s with a rendering bandwidth of 512 kbit/s. Where the region fill flag is set, the region fill in the subtitle decoder model happens before any objects are rendered into the region.

Regions are only filled when the region fill flag is set. There is no fill operation when a region is introduced at the start of an epoch. This allows the encoder to defer the fill operation, and hence the rendering burden until later.

A decoder can optionally look at the intersection between the objects in the region's object list and the area to be filled and then only fill the area not covered by objects. Decoders should take into account that objects can have a ragged right hand edge and can contain transparent holes. Any such optimization is beyond the scope of the subtitle decoder model.

5.4.4 CLUT modification

Once introduced a region is always bound to a particular CLUT. However, new definitions of the CLUT may be broadcast, i.e. the mapping between pixel code and displayed colour can be redefined. No rendering burden is assumed when CLUT definitions change.

5.4.5 Graphic object decoding

Graphical objects shall be rendered into the pixel buffer as they are decoded. One object may be referenced several times, for example, a character used several times in a piece of text. Within a region the rendering burden for each object is derived from:

- the number of pixels enclosed within the smallest rectangle that can enclose the object;
- the pixel depth of the region where the object is positioned;
- the number of times the object is positioned in the region.

The "smallest enclosing rectangle" rule is used to simplify calculations and also to give some consideration for the read-modify-write nature of pixel rendering processes.

The object coding allows a ragged right edge to objects. No coded information is provided for the pixel positions between the "end of object line code" and the "smallest enclosing rectangle" and therefore these pixels should be left unmodified by the rendering process.

The same rendering burden is assumed, regardless of whether an object has the non_modifying_colour_flag set to implement holes in the object. Again this gives some consideration for the read-modify-write nature of pixel rendering processes.

5.4.6 Character object decoding

The subtitling system allows character references to be delivered as an alternative to graphical objects. The information inside such a subtitling stream is not sufficient to make such a character coded system work reliably.

A local agreement between broadcasters and equipment manufacturers may be an appropriate way to ensure reliable operation of character coded subtitles. A local agreement would probably define the characteristics of the font (character size and other metrics). It should also define a model for rendering of the characters.

6 Subtitle stream carriage in the PES layer and in the Transport Stream, and signalling

6.1 General

A DVB subtitle stream shall be carried within **PES packets** in the **MPEG-2 Transport Stream** (TS) according to ISO/IEC 13818-1 [1], in transport packets identified by the same PID.

A single subtitle stream may carry several different subtitle services. All the subtitling data required for a subtitle service shall be carried within a single subtitle stream. The different subtitle services may be subtitles in different languages for a common program. Alternatively, they may in principle be for different programs, provided that the programs share a common **PCR**.

6.2 Carriage in the PES layer

The number of segments carried in a PES packet is limited only by the maximum length of a PES packet, as defined by ISO/IEC 13818-1 [1]. The **PTS** in the PES packet header provides presentation timing information for the subtitling objects, and is associated with the subtitle data in all segments carried in that PES packet. The PTS defines the time at which the associated decoded segments should be presented. This may include removal of subtitles, for example when an entire region is removed or when all objects in a region are removed. There may be two or more PES packets with the same PTS value, for example when it is not possible or desirable to include all segments associated to the same PTS in one PES packet.

Table 2 specifies the parameters of the PES packet that shall be used to transport subtitle streams.

Table 2: PES packet carriage of subtitle streams

PES packet field	Field setting
stream_id	Set to '1011 1101' indicating "private_stream_1".
PES_packet_length	Set to a value that specifies the length of the PES packet, as defined in ISO/IEC 13818-1 [1].
data_alignment_indicator	Set to '1' indicating that the subtitle segments are aligned with the PES packets.
PTS of subtitle page	The Presentation Time Stamp, indicating the time at which the presentation begins of the display set carried by the PES packet(s) with this PTS. The PTSs of subsequent displays shall differ by more than one video frame.
PES_packet_data_byte	The PES_data_field, as specified in table 3.

When carrying a DVB subtitle stream, the PES_packet_data_bytes shall be encoded as the PES_data_field as defined in table 3.

Table 3: PES data field

Syntax	Size	Type
PES_data_field() {		
data_identifier	8	bslbf
subtitle_stream_id	8	bslbf
while (next_bits(8) == '0000 1111') {		
subtitling_segment()		
}		
end_of_PES_data_field_marker	8	bslbf
}		

Semantics:

data_identifier: For DVB subtitle streams the data_identifier field shall be coded with the value 0x20.

subtitle_stream_id: This identifies the subtitle stream in this PES packet. A DVB subtitling stream shall be identified by the value 0x00.

subtitling_segment(): One or more subtitling segments, as defined in clause 7.2, can be included in a single PES data field. Each subtitling_segment starts with the sync byte of '0000 1111'.The number of subtitling segments contained in the PES packet is not signalled explicitly.

end_of_PES_data_field_marker: An 8-bit field with fixed contents '1111 1111'.

6.3 Carriage and signalling in the transport stream

The subtitling stream PES layer shall be carried in the MPEG-2 Transport Stream as specified in ISO/IEC 13818-1 [1].

Table 4 specifies the parameters of the Transport Stream that shall be used to transport subtitle streams.

Table 4: TS carriage of subtitle streams

stream_type in the PMT	Set to '0x06' indicating "PES packets containing private data".

For each subtitle service a **subtitling_descriptor** as defined in ETSI EN 300 468 [2] shall signal the properties of the subtitle service in the PMT of the Transport Stream carrying that subtitle service.

The **subtitling_type** field in the subtitling_descriptor shall be set according to the subtitle service properties and features used in the subtitle service, as shown in table 5. The value of subtitling_type implicitly signals the version of the present document with which the subtitle service is compliant.

The subtitling_type value shall be set to the same value as the component_type value of a DVB component descriptor as defined in ETSI EN 300 468 [2] when the stream_content field of that descriptor is equal to '0x3'. Due to the evolution of the present document, features have been added to each new version. Obviously, features introduced in any version of the present document will not be supported by IRDs that were designed to be compliant with an earlier version of the specification, hence the subtitle service shall use a value of subtitling_type corresponding to the associated service, and should use only those features, i.e. segment types and ODS coding types, that were specified in the corresponding version of the present document. Subtitle services that choose not to follow this recommendation could face issues of incompatibility with legacy subtitle decoders that might not be robust against the presence of unknown or unsupported subtitling features in the subtitle service.

IRDs shall ignore subtitle services signalled with a subtitling_type that they do not support.

NOTE: It is known that some early implementations of subtitle decoders might not ignore nor be robust against the presence of unsupported subtitling_types in subtitle bitstreams.

Table 5 lists the features of the present document that are not recommended to be used in subtitle services that are provided in accordance with a particular version of the present document, which is implicitly signalled by the subtitling_type field in the subtitling_descriptor in the PMT.

Table 5: Subtitling type usage

Subtitling type in the subtitling_descriptor (see ETSI EN 300 468 [2])	ETSI EN 300 743 version compliance	Indicative service compatibility	Features that are not recommended for the subtitle service
0x10-0x13, 0x20-0x23	V1.1.1, V1.2.1	SDTV	DDS, DSS, ACS, ODS object coding type = '2'
0x14, 0x24	V1.3.1	HDTV, UHDTV [1]	DSS, ACS, ODS object coding type = '2'
0x15, 0x25	V1.4.1, V1.5.1	3DTV	ACS, ODS object coding type = '2'
0x16, 0x26	V1.6.1	HDTV [2], UHDTV	None
NOTE 1: The subtitle service may use only the CLUT definition segment (CDS) to define the available subtitle colours within the Recommendation ITU-R BT.601 [3] colour system. NOTE 2: The subtitle service may use ODS object coding type = '2' but in that case decoders compliant with V1.5.1 or earlier of the present document will not be able to decode the subtitles.			

The subtitling_descriptor shall indicate the page id values of the segments needed to decode that subtitle service. The page id of segments with data specific to that service is referred to as the **composition page id**, while the page id of segments with shared data is referred to as the **ancillary page id**.

Version 1.6.1 of the present document introduces two new features that could, in principle, also be used with non-UHDTV service types. These features are progressive-scan bitmap objects and the alternative CLUT segment. The principle of decoder compatibility implies that if the service provider intends to maintain interoperability with existing decoders supporting an earlier version of the present document, then the new features of the later version of the present document shall not be used.

In other words, a DVB service may include subtitles with capabilities signalled with a subtitling_type that indicates a lower level of indicative service compatibility than would be expected with the associated service.

For example, a UHDTV service could include subtitle streams that do not use the new features introduced in V1.6.1, and can therefore be signalled using subtitling types 0x14 and/or 0x24, if the service provider chooses to target UHDTV IRDs with subtitle decoders that are compliant with ETSI EN 300 743 (V1.3.1) [6], ETSI EN 300 743 (V1.4.1) [7] or ETSI EN 300 743 (V1.5.1) [8] of the present document. However the service provider should bear in mind that there might be unpredictable results with the positioning of such subtitles on the screen with some UHDTV IRDs.

Conversely, if a service provider wishes to deploy progressively-coded subtitles (with ODS object coding type = '2'), subtitling type 0x16 or 0x26 shall be signalled, even if the service is not a UHDTV service.

7 Subtitling service data specification

7.1 Introduction

The present clause contains the specification of the syntax and semantics of the subtitling segment, and all subtitling segment types, in clause 7.2.

Clause 7.3 contains the specification of interoperability points for subtitle services and decoders.

7.2 Syntax and semantics of the subtitling segment

7.2.0 General

7.2.0.1 Segment syntax

The basic syntactical element of subtitle streams is the "segment". It forms the common format shared amongst all elements of this subtitling specification. A segment shall be encoded as described in table 6.

Table 6: Generic subtitling segment

Syntax	Size	Type
subtitling_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
segment_data_field()		
}		

Semantics:

sync_byte: An 8-bit field that shall be coded with the value '0000 1111'. Inside a PES packet, decoders can use the sync_byte to verify synchronization when parsing segments based on the segment_length, so as to determine transport packet loss.

segment_type: This indicates the type of data contained in the segment data field. Table 7 lists the segment_type values defined in the present document. Segment types that are not recognized or supported shall be ignored, without impacting the decoding of all recognized and supported segment types contained in the subtitling PES packet.

NOTE: It is known that some early implementations of subtitle decoders might not be robust against the presence of unsupported segment types in subtitle bitstreams.

Table 7: Segment types

Value	Segment type	Cross-reference
0x10	page composition segment	defined in clause 7.2.2
0x11	region composition segment	defined in clause 7.2.3
0x12	CLUT definition segment	defined in clause 7.2.4
0x13	object data segment	defined in clause 7.2.5
0x14	display definition segment	defined in clause 7.2.1
0x15	disparity signalling segment	defined in clause 7.2.7
0x16	alternative_CLUT_segment	defined in clause 7.2.8
0x17 - 0x7F	reserved for future use	
0x80	end of display set segment	defined in clause 7.2.6
0x81 - 0xEF	private data	
0xFF	stuffing (see note)	
All other values	reserved for future use	
NOTE: The present document does not define a syntax for stuffing within the PES. In applications where stuffing is deemed to be necessary (for example for monitoring or for network management reasons) implementers of DVB subtitle coding equipment are strongly advised to use the transport packet adaptation field for stuffing since that method will usually place no processing overhead on the subtitle encoder.		

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: The segment_length shall specify the number of bytes contained in the immediately following segment_data_field.

segment_data_field: This is the payload of the segment. The syntax of this payload depends on the segment type, and is defined in clauses 7.2.1 to 7.2.8.

7.2.0.2 Forward compatibility

The segment structure allows forward compatibility with future revisions of the present document.

NOTE: IRDs are expected to be robust against new segment types that might be added in future revisions of the present document. IRDs are also expected to be robust against the backward compatible addition or extension of data structures, and the assignment of reserved element values in future revisions of the present document.

The following explicit requirement for IRD forward compatibility was added in version 1.6.1 of the present document. Thus its mandatory nature is limited to IRDs with "UHDTV" subtitling support as defined in table 35 (in clause 7.3, interoperability points). For all other IRDs, forward compatibility is recommended.

IRDs shall ignore segment types that they do not support, without impacting decoding of segment types they do support. If the IRD encounters unknown structures or reserved values within a segment, then it shall decode the parts it is able to decode, or ignore the segment.

7.2.1 Display definition segment

The display definition for a subtitle service may be defined by the display definition segment (DDS).

Absence of a DDS in the subtitle service implies that the stream is coded in accordance with ETSI EN 300 743 (V1.2.1) [5] and that a display resolution of 720 by 576 pixels may be assumed, i.e. the subtitle service is associated with an SDTV service. Such streams will nevertheless be decodable by subtitling decoders that are compliant with any later versions of the present document. Subtitle streams associated with HDTV services may include the DDS.

Subtitle streams associated with UHDTV services shall include the DDS, whereby subtitle graphics rendering shall be constrained to HDTV resolution. If no display window is signalled, then the IRD shall apply a resolution upscale of factor two in both horizontal and vertical directions when rendering the subtitles on a UHDTV resolution screen. If the display window feature is used with subtitles for a UHDTV service, then the display window shall be specified as having dimension no larger than the maximum display resolution for HDTV, i.e. 1 920 by 1 080 pixels, within the larger UHDTV display resolution, which may be any of the dimensions allowed in ETSI TS 101 154 [9], up to the maximum display resolution for UHDTV, i.e. 3 840 by 2 160 pixels. Hence with UHDTV the display_window_horizontal_position_maximum minus display_window_horizontal_position_minimum shall be no more than 1 919, and the display_window_vertical_position_maximum minus display_window_vertical_position_minimum shall be no more than 1 079. When the display window feature is used then the IRD shall not upscale the subtitle object spatially. As specified in clause 6.3, subtitle streams that are intended to be decoded by decoders that are compliant with ETSI EN 300 743 (V1.2.1) [5] shall not include a DDS.

As specified in clause 6.3, subtitle streams which include a display definition segment shall be distinguished from those that have been coded in accordance with ETSI EN 300 743 (V1.2.1) [5], by the use of HDTV-specific or UHDTV-specific **subtitling_type** values in the subtitling descriptor signalled in the PMT for that service. This provides a means whereby legacy SDTV-only decoders should ignore streams which include a display definition segment.

A subtitle stream shall not convey both a subtitle service which includes a DDS and one that does not; in this case the subtitle services shall be carried in separate streams and on separate PIDs.The syntax of the DDS is shown in table 8.

Table 8: Display definition segment

Syntax	Size	Type
display_definition_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	uimsbf
segment_length	16	uimsbf
dds_version_number	4	uimsbf
display_window_flag	1	uimsbf
reserved	3	uimsbf
display_width	16	uimsbf
display_height	16	uimsbf
if (display_window_flag == 1) {		
display_window_horizontal_position_minimum	16	uimsbf
display_window_horizontal_position_maximum	16	uimsbf
display_window_vertical_position_minimum	16	uimsbf
display_window_vertical_position_maximum	16	uimsbf
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x14, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

dds_version_number: The version of this display definition segment. When any of the contents of this display definition segment change, this version number is incremented (modulo 16).

display_window_flag: If display_window_flag = 1, the DVB subtitle display set associated with this display definition segment is intended to be rendered in a window within the display resolution defined by display_width and display_height. The size and position of this window within the display is defined by the parameters signalled in this display definition segment as display_window_horizontal_position_minimum, display_window_horizontal_position_maximum, display_window_vertical_position_minimum and display_window_vertical_position_maximum.

If display_window_flag = 0, the DVB subtitle display set associated with this display_definition_segment is intended to be rendered directly within the display resolution defined by display_width and display_height.

display_width: Specifies the maximum horizontal width of the display in pixels minus 1 assumed by the subtitling stream associated with this display definition segment. The value in this field shall be in the region 0..4095.

display_height: Specifies the maximum vertical height of the display in lines minus 1 assumed by the subtitling stream associated with this display definition segment. The value in this field shall be in the region 0..4095.

display_window_horizontal_position_minimum: Specifies the left-hand most pixel of this DVB subtitle display set with reference to the left-hand most pixel of the display.

display_window_horizontal_position_maximum: Specifies the right-hand most pixel of this DVB subtitle display set with reference to the left-hand most pixel of the display.

display_window_vertical_position_minimum: Specifies the upper most line of this DVB subtitle display set with reference to the top line of the display.

display_window_vertical_position_maximum: Specifies the bottom line of this DVB subtitle display set with reference to the top line of the display.

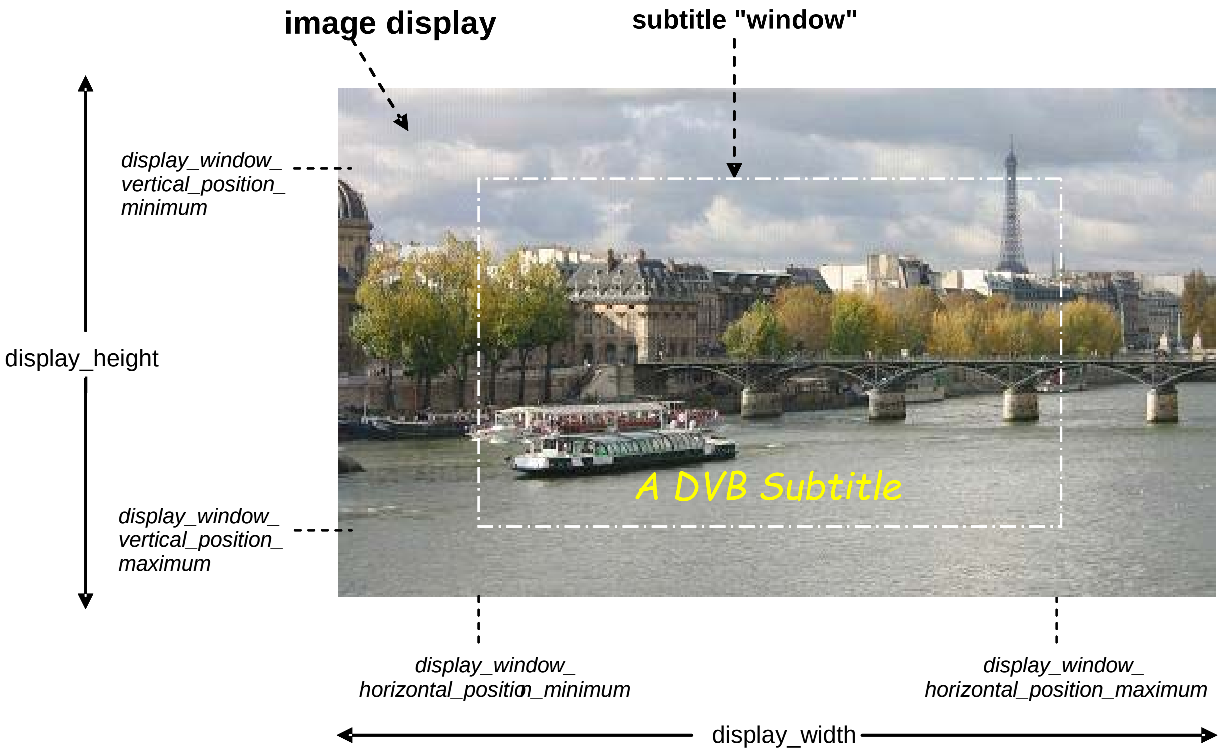


Figure 4: Use of Display definition segment parameters

HDTV and UHDTV IRDs that offer a means of scaling or positioning the subtitles under user control (e.g. to make them larger or smaller) can use the information conveyed in the display definition segment to determine safe strategies for zooming and/or positioning that will ensure that windowed subtitles can remain visible. However, scaling operations are not recommended for subtitles that have been anti-aliased for their original graphical resolution. Any scaling applied to such subtitles could degrade them significantly and thereby impact their readability.

7.2.2 Page composition segment

The page composition for a subtitle service is carried in page_composition_segments. The page_id of each page_composition_segment shall be equal to the composition_page_id value provided by the subtitling descriptor.

The syntax of the page_composition_segment is shown in table 9.

Table 9: Page composition segment

Syntax	Size	Type
page_composition_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
page_time_out	8	uimsbf
page_version_number	4	uimsbf
page_state	2	bslbf
reserved	2	bslbf
while (processed_length < segment_length) {		
region_id	8	bslbf
reserved	8	bslbf
region_horizontal_address	16	uimsbf
region_vertical_address	16	uimsbf
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x10, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

page_time_out: The period, expressed in seconds, after which a page instance is no longer valid and consequently shall be erased from the screen, should it not have been redefined before that. The time-out period starts when the page instance is first displayed. The page_time_out value applies to each page instance until its value is redefined. The purpose of the time-out period is to avoid a page instance remaining on the screen "for ever" if the IRD happens to have missed the redefinition or deletion of the page instance. The time-out period does not need to be counted very accurately by the IRD: a reaction accuracy of -0/+5 s is accurate enough.

page_version_number: The version of this page composition segment. When any of the contents of this page composition segment change, this version number is incremented (modulo 16).

page_state: This field signals the status of the subtitling page instance described in this page composition segment. The values of the page_state are defined in table 10.

Table 10: Page state

Value	Page state	Effect on page	Comments
00	normal case	page update	The display set contains only the subtitle elements that are changed from the previous page instance.
01	acquisition point	page refresh	The display set contains all subtitle elements needed to display the next page instance.
10	mode change	new page	The display set contains all subtitle elements needed to display the new page.
11	reserved		Reserved for future use.

If the page state is "mode change" or "acquisition point", then the display set shall contain a region composition segment for each region used in this epoch.

processed_length: The total number of bytes that have already been processed following the segment_length field.

region_id: This uniquely identifies a region within a page. Each identified region is displayed in the page instance defined in this page composition. Regions shall be listed in the page_composition_segment in the order of ascending region_vertical_address field values.

region_horizontal_address: This specifies the horizontal address of the top left pixel of this region. The left-most pixel of the active pixels has horizontal address zero, and the pixel address increases from left to right.

region_vertical_address: This specifies the vertical address of the top line of this region. The top line of the frame is line zero, and the line address increases by one within the frame from top to bottom.

NOTE: All addressing of pixels is based on a frame of M pixels horizontally by N scan lines vertically. These numbers are independent of the aspect ratio of the picture; on a 16:9 display a pixel looks a bit wider than on a 4:3 display. In some cases, for instance a logo, this may lead to unacceptable distortion. Separate data may be provided for presentation on each of the different aspect ratios. The subtitling descriptor signals whether the associated subtitle data can be presented on any display or on displays of specific aspect ratio only.

7.2.3 Region composition segment

The region composition for a specific region is carried in region_composition_segments. The region composition contains a list of objects; the listed objects shall be positioned in such a way that they do not overlap.

If an object is added to a region in case of a page update, new pixel data will overwrite either the background colour of the region or "old objects". The programme provider shall take care that the new pixel data overwrites only information that needs to be replaced, but also that it overwrites all pixels in the region that are not to be preserved. Note that a pixel is either defined by the background colour, or by an "old" object or by a "new" object; if a pixel is overwritten none of its previous definition is retained.

Table 11 shows the syntax of the region composition segment.

Table 11: Region composition segment

Syntax	Size	Type
region_composition_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
region_id	8	uimsbf
region_version_number	4	uimsbf
region_fill_flag	1	bslbf
reserved	3	bslbf
region_width	16	uimsbf
region_height	16	uimsbf
region_level_of_compatibility	3	bslbf
region_depth	3	bslbf
reserved	2	bslbf
CLUT_id	8	bslbf
region_8-bit_pixel_code	8	bslbf
region_4-bit_pixel-code	4	bslbf
region_2-bit_pixel-code	2	bslbf
reserved	2	bslbf
while (processed_length < segment_length) {		
object_id	16	bslbf
object_type	2	bslbf
object_provider_flag	2	bslbf
object_horizontal_position	12	uimsbf
reserved	4	bslbf
object_vertical_position	12	uimsbf
if (object_type ==0x01 or object_type == 0x02) {		
foreground_pixel_code	8	bslbf
background_pixel_code	8	bslbf
}		
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x11, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

region_id: This 8-bit field uniquely identifies the region for which information is contained in this region_composition_segment.

region_version_number: This indicates the version of this region. The version number is incremented (modulo 16) if one or more of the following conditions is true:

- the region_fill_flag is set;
- the region's CLUT family has been modified;
- the region has a non-zero length object list.

region_fill_flag: If set to '1', signals that the region is to be filled with the background colour defined in the region_n-bit_pixel_code fields in this segment.

region_width: Specifies the horizontal length of this region, expressed in number of pixels. For subtitle services which do not include a display definition segment, the value in this field shall be within the range 1 to 720, and the sum of the region_width and the region_horizontal_address (see clause 7.2.1) shall not exceed 720. For subtitle services which include a display definition segment, the value of this field shall be within the range 1 to (display_width +1) and shall not exceed the value of (display_width +1) as signalled in the relevant DDS.

region_height: Specifies the vertical length of the region, expressed in number of pixels. For subtitle services which do not include a display definition segment, the value in this field shall be within the inclusive range 1 to 576, and the sum of the region_height and the region_vertical_address (see clause 7.2.1) shall not exceed 576. For subtitle services which include a display definition segment, the value of this field shall be within the range 1 to (display_height +1) and shall not exceed the value of (display_height +1) as signalled in the relevant DDS.

region_level_of_compatibility: This indicates the minimum type of CLUT that is necessary in the decoder to decode this region as defined in table 12.

Table 12: Region level of compatibility

Value	Minimum CLUT type
0x00	reserved
0x01	2-bit/entry CLUT required
0x02	4-bit/entry CLUT required
0x03	8-bit/entry CLUT required
0x04..0x07	reserved

If the decoder does not support the specified minimum requirement for the type of CLUT, then this region shall not be displayed, even though some other regions, requiring a lesser type of CLUT, may be presented.

region_depth: Identifies the intended pixel depth for this region as defined in table 13.

Table 13: Intended region pixel depth

Value	Intended region pixel depth
0x00	reserved
0x01	2 bit
0x02	4 bit
0x03	8 bit
0x04..0x07	reserved

CLUT_id: Identifies the family of CLUTs that applies to this region.

region_8-bit_pixel-code: Specifies the entry of the applied 8-bit CLUT as background colour for the region when the region_fill_flag is set, but only if the region depth is 8 bit. The value of this field is undefined if a region depth of 2 or 4 bit applies.

region_4-bit_pixel-code: Specifies the entry of the applied 4-bit CLUT as background colour for the region when the region_fill_flag is set, if the region depth is 4 bit, or if the region depth is 8 bit while the region_level_of_compatibility specifies that a 4-bit CLUT is within the minimum requirements. In any other case the value of this field is undefined.

region_2-bit_pixel-code: Specifies the entry of the applied 2-bit CLUT as background colour for the region when the region_fill_flag is set, if the region depth is 2 bit, or if the region depth is 4 or 8 bit while the region_level_of_compatibility specifies that a 2-bit CLUT is within the minimum requirements. In any other case the value of this field is undefined.

processed_length: The total number of bytes that have already been processed following the segment_length field.

object_id: Identifies an object that is shown in the region.

object_type: Identifies the type of object as defined in table 14.

Table 14: Object type

Value	Object type
0x00	basic_object, bitmap
0x01	basic_object, character
0x02	composite_object, string of characters
0x03	reserved

object_provider_flag: A 2-bit flag indicating how this object is provided, as defined in table 15.

Table 15: Object provider flag

Value	Object provision
0x00	provided in the subtitling stream
0x01	provided by a ROM in the IRD
0x02	reserved
0x03	reserved

object_horizontal_position: Specifies the horizontal position of the top left pixel of this object, expressed in number of horizontal pixels, relative to the left-hand edge of the associated region. The specified horizontal position shall be within the region, hence its value shall be in the range between 0 and (region_width -1).

object_vertical_position: Specifies the vertical position of the top left pixel of this object, expressed in number of lines, relative to the top of the associated region. The specified vertical position shall be within the region, hence its value shall be in the range between 0 and (region_height -1).

foreground_pixel_code: Specifies the entry in the applied 8-bit CLUT that has been selected as the foreground colour of the character(s).

background_pixel_code: Specifies the entry in the applied 8-bit CLUT that has been selected as the background colour of the character(s).

NOTE: IRDs with CLUT of four or sixteen entries find the foreground and background colours through the reduction schemes described in clause 9.

7.2.4 CLUT definition segment

The CLUT definition segment signals modifications to one or more CLUTs within a particular CLUT family. The modifications define replacement Recommendation ITU-R BT.601 [3] colours that can selectively modify one or more entries by replacing the default initial values (defined in clause 10). A subtitle service can thus create and use a CLUT consisting of a combination of colours in the default CLUT and colours not contained in the default CLUT. The segment syntax is defined in table 16.

For the purpose of backward compatibility of subtitle services with existing decoders, subtitle services shall support rendering in the Recommendation ITU-R BT.601 [3] colour space, via provision of the CDS, if not relying on the default CLUTs. This shall be the case even when the subtitle service makes use of the alternative_CLUT_segment (ACS) (defined in clause 7.2.8). However, in this case, for each ACS, a CDS with the same CLUT_id shall contain an entry for each of the colours used, using the 8-bits per entry option only, i.e. with the 8-bits per entry flag set to '1'. Each colour in the CDS shall be a colour within the Recommendation ITU-R BT.601 [3] colour space that is a close equivalent to the corresponding colour defined in the ACS.

The 8-bit CLUT entry format allows a sufficient number of colours to be used in order to achieve high quality anti-aliasing. This mitigates the effects of spatial upscaling, especially with UHDTV services. For the same reason, also when only the CDS is used with UHDTV services (i.e. no ACS is provided), it is recommended to use the 8-bit CLUT entry form of the CDS.

Table 16: CLUT definition segment

Syntax	Size	Type
CLUT_definition_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
CLUT_id	8	bslbf
CLUT_version_number	4	uimsbf
reserved	4	bslbf
while (processed_length < segment_length) {		
CLUT_entry_id	8	bslbf
2-bit/entry_CLUT_flag	1	bslbf
4-bit/entry_CLUT_flag	1	bslbf
8-bit/entry_CLUT_flag	1	bslbf
reserved	4	bslbf
full_range_flag	1	bslbf
if full_range_flag =='1' {		
Y-value	8	bslbf
Cr-value	8	bslbf
Cb-value	8	bslbf
T-value	8	bslbf
} else {		
Y-value	6	bslbf
Cr-value	4	bslbf
Cb-value	4	bslbf
T-value	2	bslbf
}		
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x12, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

CLUT_id: Uniquely identifies within a page the CLUT family whose data is contained in this CLUT_definition_segment field.

CLUT_version_number: Indicates the version of this segment data. When any of the contents of this segment change this version number is incremented (modulo 16).

processed_length: The total number of bytes that have already been processed following the segment_length field.

CLUT_entry_id: Specifies the entry number of the CLUT. The first entry of the CLUT has entry number zero.

2-bit/entry_CLUT_flag: If set to '1', this indicates that this CLUT value is to be loaded into the identified entry of the 2-bit/entry CLUT. This option shall not be used when the CDS accompanies an alternative CLUT segment (ACS).

4-bit/entry_CLUT_flag: If set to '1', this indicates that this CLUT value is to be loaded into the identified entry of the 4-bit/entry CLUT. This option shall not be used when the CDS accompanies an alternative CLUT segment (ACS).

8-bit/entry_CLUT_flag: If set to '1', this indicates that this CLUT value is to be loaded into the identified entry of the 8-bit/entry CLUT. This option shall be used when the CDS accompanies an alternative CLUT segment (ACS).

Only one N-bit/entry_CLUT_flag shall be set to 1 per CLUT_entry_id and its associated Y-, Cr-, Cb- and T-values.

full_range_flag: If set to '1', this indicates that the Y_value, Cr_value, Cb_value and T_value fields have the full 8-bit resolution. If set to '0', then these fields contain only the most significant bits.

Y_value: The Y output value of the CLUT for this entry. A value of zero in the Y_value field signals full transparency. In that case the values in the Cr_value, Cb_value and T_value fields are irrelevant and shall be set to zero.

NOTE 1: Implementers should note that Y=0 is disallowed in Recommendation ITU-R BT.601 [3]. This condition should be recognized and mapped to a legal value (e.g. Y=16d) before conversion to RGB values in a decoder.

Cr_value: The Cr output value of the CLUT for this entry.

Cb_value: The Cb output value of the CLUT for this entry.

NOTE 2: Y, Cr and Cb have meanings as defined in Recommendation ITU-R BT.601 [3] and in Recommendation ITU-R BT.656-4 [4].

NOTE 3: Note that, whilst this subtitling specification defines CLUT entries in terms of Y, Cr, Cb and T values, the standard interface definition of digital television (Recommendation ITU-R BT.656-4 [4]) presents co-sited sample values in the order Cb,Y,Cr. Failure to correctly interpret the rendered bitmap image in terms of Recommendation ITU-R BT.656-4 [4] may result in incorrect colours and chrominance mistiming.

T_value: The Transparency output value of the CLUT for this entry. A value of zero identifies no transparency. The maximum value plus one would correspond to full transparency. For all other values the level of transparency is defined by linear interpolation.

Full transparency is acquired through a value of zero in the Y_value field.

NOTE 4: Decoder models for the translation of pixel-codes into Y, Cr, Cb and T values are depicted in clause 9. Default contents of the CLUT are specified in clause 10.

NOTE 5: The colour for each CLUT entry can be redefined. There is no need for CLUTs with fixed contents as every CLUT has default contents, see clause 10.

7.2.5 Object data segment

7.2.5.0 General

The object_data_segment contains the data of an object. For graphical objects with the object_coding_method setting of "coding of pixels" the following applies:

- an object may be interlaced, with a top field and a bottom field or a top field that is repeated as the bottom field, or it may be progressive, with a single field of object data;
- the first pixel of the first line of the top field is the top left pixel of the object;
- the first pixel of the first line of the bottom field is the most left pixel on the second line of the object;
- for interlaced objects:
  - the same object_data_segment shall carry a pixel-data_sub-block for both the top field and the bottom field;
  - if a segment carries no data for the bottom field, i.e. the bottom_field_data_block_length contains the value '0x0000', then the pixel-data_sub-block for the top field shall apply for the bottom field also.

The object_data_segment is defined as shown in table 17.

Table 17: Object data segment

Syntax	Size	Type
object_data_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
object_id	16	bslbf
object_version_number	4	uimsbf
object_coding_method	2	bslbf
non_modifying_colour_flag	1	bslbf
reserved	1	bslbf
if (object_coding_method == '00'){		
top_field_data_block_length	16	uimsbf
bottom_field_data_block_length	16	uimsbf
while(processed_length<top_field_data_block_length)		
pixel-data_sub-block()		
while (processed_length<bottom_field_data_block_length)		
pixel-data_sub-block()		
if (stuffing_length == 1)		
8_stuff_bits	8	bslbf
}		
if (object_coding_method == '01') {		
number of codes	8	uimsbf
for (i == 1, i <= number of codes, i ++)		
character_code	16	bslbf
}		
if (object_coding_method == '10'){		
progressive_pixel_block()		
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x13, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

object_id: Uniquely identifies within the page the object for which data is contained in this object_data_segment field.

object_version_number: Indicates the version of this segment data. When any of the contents of this segment change, this version number is incremented (modulo 16).

object_coding_method: Specifies the method used to code the object, as defined in table 18.

Table 18: Object coding method

Value	Object coding method
0x0	coding of pixels (see note 1)
0x1	coded as a string of characters
0x2	progressive coding of pixels (see note 2)
0x3	reserved
NOTE 1: The value 0x0 indicates interlaced coding of pixels, the only method available for coding of pixels prior to version V1.6.1 of the present document. NOTE 2: This object coding method is introduced in version 1.6.1 of the present document, hence subtitle decoders that are compliant with an earlier version of the present document will be unable to process this mode.	

non_modifying_colour_flag: If set to '1' this indicates that the CLUT entry value '1' is a non modifying colour. When the non modifying colour is assigned to an object pixel, then the pixel of the underlying region background or object shall not be modified. This can be used to create "transparent holes" in objects.

top_field_data_block_length: Specifies the number of bytes contained in the pixel-data_sub-blocks for the top field.

bottom_field_data_block_length: Specifies the number of bytes contained in the data_sub-block for the bottom field.

pixel-data_sub-block(): Contains the run-length encoded data for each field of the object. Its structure is defined in clause 7.2.5.1.

processed_length: The number of bytes from the field(s) within the while-loop that have been processed by the decoder.

stuffing_length: The value is not signalled but it can be calculated from other fields and shall be either zero or one.

NOTE: In earlier versions of the present document the presence or absence of the 8_stuff_bits field was determined by an undefined wordaligned() function which created an ambiguity. This was replaced by the stuffing_length value to remove the ambiguity.

Some legacy subtitle encoders may operate differently to the recommended behaviour defined below in table 19. However in all cases subtitle decoders shall calculate the stuffing_length value using the following equation:

stuffing_length = segment_length - 7 - top_field_data_block_length - bottom_field_data_block_length

Subtitle encoders should add an 8_stuff_bits field only if the sum of top_field_data_block_length and bottom_field_data_block_length is an even number. Therefore the segment_length field will always be set to an even number. The recommended encoder behaviour is summarized in table 19.

Table 19: Recommended encoding of object_data_segment

top_field_data_block_length + bottom_field_data_block_length	8_stuff_bits	stuffing_length (implied)	segment_length
Is an Odd number	Not present	0	7 + stuffing_length + top_field_data_block_length + bottom_field_data_block_length (*Always an even number*)
Is an Even number	Present	1	

8_stuff_bits: If present, this field shall be coded as '0000 0000'.

number_of_codes: Specifies the number of character codes in the string.

character_code: Specifies a character through its index number in a character table, the definition of which is not included in the present document. The specification and provision of such a character code table is part of the local agreement between the subtitle service provider and IRD manufacturer that is needed to put this mode of subtitles into operation.

progressive_pixel_block(): Contains the data for the progressively coded object. Its structure is defined in clause 7.2.5.3.

7.2.5.1 Pixel-data sub-block

The pixel-data sub-block structure is used with object coding method 0x0, i.e. "coding of pixels".

For each object the pixel-data sub-block for the top field and the pixel-data sub-block for the bottom field shall be carried in the same object_data_segment. If this segment carries no data for the bottom field, i.e. the bottom_field_data_block_length contains the value '0x0000', then the data for the top field shall be valid for the bottom field also.

NOTE: This effectively forbids an object from having a height of only one TV picture line. Isolated objects of this height would be liable to suffer unpleasant flicker effects at the TV display frame rate when displayed on an interlaced display.

Table 20 defines the syntax of the pixel-data sub-block structure.

Table 20: Pixel-data sub-block

Syntax	Size	Type
pixel-data_sub-block() {		
data_type	8	bslbf
if data_type =='0x10' {		
repeat {		
2-bit/pixel_code_string()		
} until (end of 2-bit/pixel_code_string)		
while (!bytealigned())		
2_stuff_bits	2	bslbf
if data_type =='0x11' {		
repeat {		
4-bit/pixel_code_string()		
} until (end of 4-bit/pixel_code_string)		
if (!bytealigned())		
4_stuff_bits	4	bslbf
}		
}		
if data_type =='0x12' {		
repeat {		
8-bit/pixel_code_string()		
} until (end of 8-bit/pixel_code_string)		
}		
if data_type =='0x20'		
2_to_4-bit_map-table	16	bslbf
if data_type =='0x21'		
2_to_8-bit_map-table	32	bslbf
if data_type =='0x22'		
4_to_8-bit_map-table	128	bslbf
}		

Semantics:

data_type: Identifies the type of information contained in the pixel-data_sub-block according to table 21.

Table 21: Data type

Value	data_type
0x10	2-bit/pixel code string
0x11	4-bit/pixel code string
0x12	8-bit/pixel code string
0x20	2_to_4-bit_map-table data
0x21	2_to_8-bit_map-table data
0x22	4_to_8-bit_map-table data
0xF0	end of object line code
NOTE: All other values are reserved.	

The data types 2-bit/pixel code string, 4-bit/pixel code string, and 8-bit/pixel code string are defined in clause 7.2.5.2.

A code '0xF0' = "end of object line code" shall be included after every series of code strings that together represent one line of the object.

2_to_4-bit_map-table: Specifies how to map the 2-bit/pixel codes on a 4-bit/entry CLUT by listing the 4 entry numbers of 4-bits each; entry number 0 first, entry number 3 last.

2_to_8-bit_map-table: Specifies how to map the 2-bit/pixel codes on an 8-bit/entry CLUT by listing the 4 entry numbers of 8-bits each; entry number 0 first, entry number 3 last.

4_to_8-bit_map-table: Specifies how to map the 4-bit/pixel codes on an 8-bit/entry CLUT by listing the 16 entry numbers of 8-bits each; entry number 0 first, entry number 15 last.

2_stuff_bits: Two stuffing bits that shall be coded as '00'.

4_stuff_bits: Four stuffing bits that shall be coded as '0000'.

bytealigned(): function is true if current position is aligned to whole byte boundary from the start of the pixel-data_sub-block().

7.2.5.2 Syntax and semantics of the pixel code strings

7.2.5.2.1 2-bits per pixel code

Table 22 defines the syntax of the 2-bits per pixel code string.

Table 22: 2-bits per pixel code string

Syntax	Size	Type
2-bit/pixel_code_string() {		
if (next_bits(2) != '00') {		
2-bit_pixel-code	2	bslbf
} else {		
2-bit_zero	2	bslbf
switch_1	1	bslbf
if (switch_1 == '1') {		
run_length_3-10	3	uimsbf
2-bit_pixel-code	2	bslbf
} else {		
switch_2	1	bslbf
if (switch_2 == '0') {		
switch_3	2	bslbf
if (switch_3 == '10') {		
run_length_12-27	4	uimsbf
2-bit_pixel-code	2	bslbf
}		
if (switch_3 == '11') {		
run_length_29-284	8	uimsbf
2-bit_pixel-code	2	bslbf
}		
}		
}		
}		
}		

Semantics:

2-bit_pixel-code: A 2-bit code, specifying the pseudo-colour of a pixel as either an entry number of a CLUT with four entries or an entry number of a map-table.

2-bit_zero: A 2-bit field filled with '00'.

switch_1: A 1-bit switch that identifies the meaning of the following fields.

run_length_3-10: Number of pixels minus 3 that shall be set to the pseudo-colour defined next.

switch_2: A 1-bit switch. If set to '1', it signals that one pixel shall be set to pseudo-colour (entry) '00', else it indicates the presence of the following fields.

switch_3: A 2-bit switch that may signal one of the properties listed in table 23.

Table 23: switch_3 for 2-bits per pixel code

Value	Meaning
00	end of 2-bit/pixel_code_string
01	two pixels shall be set to pseudo colour (entry) '00'
10	the following 6 bits contain run length coded pixel data
11	the following 10 bits contain run length coded pixel data

run_length_12-27: Number of pixels minus 12 that shall be set to the pseudo-colour defined next.

run_length_29-284: Number of pixels minus 29 that shall be set to the pseudo-colour defined next.

7.2.5.2.2 4-bits per pixel code

Table 24 defines the syntax of the 4-bits per pixel code string.

Table 24: 4-bits per pixel code string

Syntax	Size	Type
4-bit/pixel_code_string() {		
if (next_bits(4) != '0000') {		
4-bit_pixel-code	4	bslbf
} else {		
4-bit_zero	4	bslbf
switch_1	1	bslbf
if (switch_1 == '0') {		
if (next_bits(3) != '000')		
run_length_3-9	3	uimsbf
Else		
end_of_string_signal	3	bslbf
} else {		
switch_2	1	bslbf
if (switch_2 == '0') {		
run_length_4-7	2	bslbf
4-bit_pixel-code	4	bslbf
} else {		
switch_3	2	bslbf
if (switch_3 == '10') {		
run_length_9-24	4	uimsbf
4-bit_pixel-code	4	bslbf
}		
if (switch_3 == '11') {		
run_length_25-280	8	uimsbf
4-bit_pixel-code	4	bslbf
}		
}		
}		
}		
}		

Semantics:

4-bit_pixel-code: A 4-bit code, specifying the pseudo-colour of a pixel as either an entry number of a CLUT with sixteen entries or an entry number of a map-table.

4-bit_zero: A 4-bit field filled with '0000'.

switch_1: A 1-bit switch that identifies the meaning of the following fields.

run_length_3-9: Number of pixels minus 2 that shall be set to pseudo-colour (entry) '0000'.

end_of_string_signal: A 3-bit field filled with '000'. The presence of this field, i.e. next_bits(3) == '000', signals the end of the 4-bit/pixel_code_string.

switch_2: A 1-bit switch. If set to '0', it signals that that the following 6-bits contain run-length coded pixel-data, else it indicates the presence of the following fields.

switch_3: A 2-bit switch that may signal one of the properties listed in table 25.

Table 25: switch_3 for 4-bits per pixel code

Value	Meaning
00	1 pixel shall be set to pseudo-colour (entry) '0000'
01	2 pixels shall be set to pseudo-colour (entry) '0000'
10	the following 8 bits contain run-length coded pixel-data
11	the following 12 bits contain run-length coded pixel-data

run_length_4-7: Number of pixels minus 4 that shall be set to the pseudo-colour defined next.

run_length_9-24: Number of pixels minus 9 that shall be set to the pseudo-colour defined next.

run_length_25-280: Number of pixels minus 25 that shall be set to the pseudo-colour defined next.

7.2.5.2.3 8-bits per pixel code

Table 26 defines the syntax of the 8-bits per pixel code string.

Table 26: 8-bits per pixel code string

Syntax	Size	Type
8-bit/pixel_code_string() {		
if (next_bits(8) != '0000 0000') {		
8-bit_pixel-code	8	bslbf
} else {		
8-bit_zero	8	bslbf
switch_1	1	bslbf
if switch_1 == '0' {		
if next_bits(7) != '000 0000'		
run_length_1-127	7	uimsbf
else		
end_of_string_signal	7	bslbf
} else {		
run_length_3-127	7	uimsbf
8-bit_pixel-code	8	bslbf
}		
}		
}		

Semantics:

8-bit_pixel-code: An 8-bit code, specifying the pseudo-colour of a pixel as an entry number of a CLUT with 256 entries.

8-bit_zero: An 8-bit field filled with '0000 0000'.

switch_1: A 1-bit switch that identifies the meaning of the following fields.

run_length_1-127: Number of pixels that shall be set to pseudo-colour (entry) '0x00'.

end_of_string_signal: A 7-bit field filled with '000 0000'. The presence of this field, i.e. next_bits(7) == '000 0000', signals the end of the 8-bit/pixel_code_string.

run_length_3-127: Number of pixels that shall be set to the pseudo-colour defined next. This field shall not have a value of less than three.

7.2.5.3 Progressive pixel block

The progressive pixel block format is used with object coding method 0x2, i.e. "progressive coding of pixels".

This object coding method is introduced in V1.6.1 of the present document, hence it shall not be used in systems where subtitle decoders are in operation that were designed to be compliant with ETSI EN 300 743 (V1.5.1) [8] or an earlier version.

Subtitle streams with progressive object coding type shall use subtitling_type value 0x16 or 0x26 in the subtitling descriptor signalled in the PMT for the service in which they are carried. Subtitle streams that have subtitling_type value not equal to either 0x16 or 0x26 shall not use the progressive coding object type. This ensures that IRDs that are compliant with V1.5.1 or an earlier version of the present document should not be presented with subtitle services that use object coding method 0x2.

The progressive pixel block format shall not be used to carry interlace-scan subtitle segments.

Progressively coded subtitle bitmaps shall be carried in the *zlib* datastream format, as defined in IETF RFC 1950 [14]. This format applies the *DEFLATE* compression method as defined by IETF RFC 1951 [15]. The parameters for *zlib* and *DEFLATE* usage shall be the same as those applied in the Portable Network Graphics (PNG) format [16] with "Compression method 0" applied to the sequence of filtered scanlines, without any further PNG format overhead, i.e. without the PNG "chunk" structure.

The syntax of the progressive pixel block is shown in table 27.

Table 27: Progressive pixel block

Syntax	Size	Type
progressive_pixel_block() {		
bitmap_width	16	uimsbf
bitmap_height	16	uimsbf
compressed_data_block_length	16	uimsbf
for (i=0; i<compressed_data_block_length; i++) {		
compressed_bitmap_data_byte	8	bslbf
}		
}		

Semantics:

bitmap_width: This field shall indicate the width of the subtitle bitmap image in pixels.

bitmap_height: This field shall indicate the height of the subtitle bitmap image in pixels.

compressed_data_block_length: This field shall indicate the number of *compressed_bitmap_data_byte* following this field.

compressed_bitmap_data_byte: This field is formed of the sequence of bytes of the subtitle bitmap image in compressed form, which is according to the *zlib* container format [14], in the same way as is specified for the Portable Network Graphics (PNG) format [16]. This format applies the *DEFLATE* compression algorithm [15]. The compressed bitmap data shall consist of the raw *zlib* datastream and shall not contain any PNG format overhead such as chunk headers or chunk CRC values.

Annex E provides an informative description of the conversion process for a suitably coded PNG file to be converted into a progressively-coded subtitle bitmap.

7.2.6 End of display set segment

The end_of_display_set_segment provides an explicit indication to the decoder that transmission of a display set is complete. The end_of_display_set_segment shall be inserted into the stream as the last segment for each display set. It shall be present for each subtitle service in a subtitle stream, although decoders need not take advantage of this segment and may apply other strategies to determine when they have sufficient information from a display set to commence decoding. The syntax of the end_of_display_set_segment is shown in table 28.

Table 28: End of display set segment

Syntax	Size	Type
end_of_display_set_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x80, as listed in table 7.

page_id: If the subtitle service uses shared data, then the page_id shall be coded with the ancillary page id value signalled in the subtitling descriptor. Otherwise the page_id shall have the value of the composition page id.

segment_length: This field shall be set to the value zero.

7.2.7 Disparity Signalling Segment

The Disparity Signalling Segment (DSS) supports the subtitling of plano-stereoscopic 3DTV content by allowing disparity values to be ascribed to a region or to part of a region. Whilst regions cannot themselves share scan lines the DSS defines subregions which may be assigned different individual disparity values.

Absence of a DSS implies that the stream has been coded in accordance with ETSI EN 300 743 (V1.3.1) [6] to provide subtitles intended for 2D presentation. In such cases decoders capable of supporting 3D services shall apply an implicit disparity of zero.

Each region can contain one or more subregions referenced to that region. Subregions have the same height as their region and may not overlap horizontally (see figures 5 and 6).

There shall be no more than 4 subregions per region and no more than 4 subregions per display set.

A subregion shall enclose all the objects for which it conveys a particular disparity value and all objects shall be enclosed by one of the subregions of a region. All active subregions in a declared display set shall be signalled in the DSS.

A change to any data (e.g. disparity values) signalled in the DSS requires a change to the DSS version number but *does not* require a change to the version number of the RCSs nor the retransmission of the RCS if the relevant region definition itself remains unchanged.

Disparity is the difference between the horizontal positions of a pixel representing the same point in space in the right and left views of a plano-stereoscopic image. Positive disparity values move the subtitle objects enclosed by a subregion away from the viewer whilst negative values move them towards the viewer. A value of zero places the objects enclosed by that subregion in the plane of the display screen.

To ensure that subtitles are placed at the correct depth and horizontal location the disparity shift values signalled shall be applied symmetrically to each view of any subregion and by implication any object bounded by the subregion. A **positive** disparity shift value for example of +7 will result in a shift of 7 pixels to the left in the left subtitle subregion image and a shift of 7 pixels to the right in the right subtitle subregion image. A **negative** disparity shift value of -7 will result in a shift of 7 pixels to the **right** in the left subtitle subregion image and a shift of 7 pixels to the **left** in the right subtitle subregion image. Note that the actual disparity of the displayed subtitle is therefore double the value of the disparity shift values signalled in the disparity integer and/or fractional fields carried in the DSS.

Encoders shall assign a value of disparity to the default disparity (and its associated disparity_update_sequence if present) which would result in an appropriate placement of the subtitles were a decoder only able to apply the default disparity to the entire display set at that time. Decoders which can support only one value of disparity per page shall apply the default disparity value to each region.

Decoders which can attribute a separate disparity value to each region (or subregion) shall parse the region loop in the DSS syntax and implement the signalled disparity shift values for the declared regions or subregions.

Encoders shall ensure that the relative position and size of multiple subregions are managed so as to avoid horizontal overlap when the objects enclosed within those subregions have the relevant disparity values applied as a shift by the decoder**.** In the event, however, that a decoder is presented with subregions whose views do overlap, the decoder should manage occlusion appropriately (for example by presenting those subregions in depth-order of perceived proximity to the viewer i.e. the foremost shown in its entirety).

Encoders that are generating streams which include a DSS shall encode the background of a region using the region fill mechanism ***only*** if the region contains a single subregion ***or*** if the region fill indexes a fully transparent CLUT entry.

A stream with a DSS shall include a Display Definition Segment and the display window parameters of that DDS shall be consistent with the application of the disparity values signalled in the DSS.

In the transmission of a display set (new or updated) the DSS will normally follow the RCS. However, if the PCS has page_state = normal and if the only changes to be signalled are disparity values, these values may be updated by the simple transmission of a DDS, a DSS and an EDS.

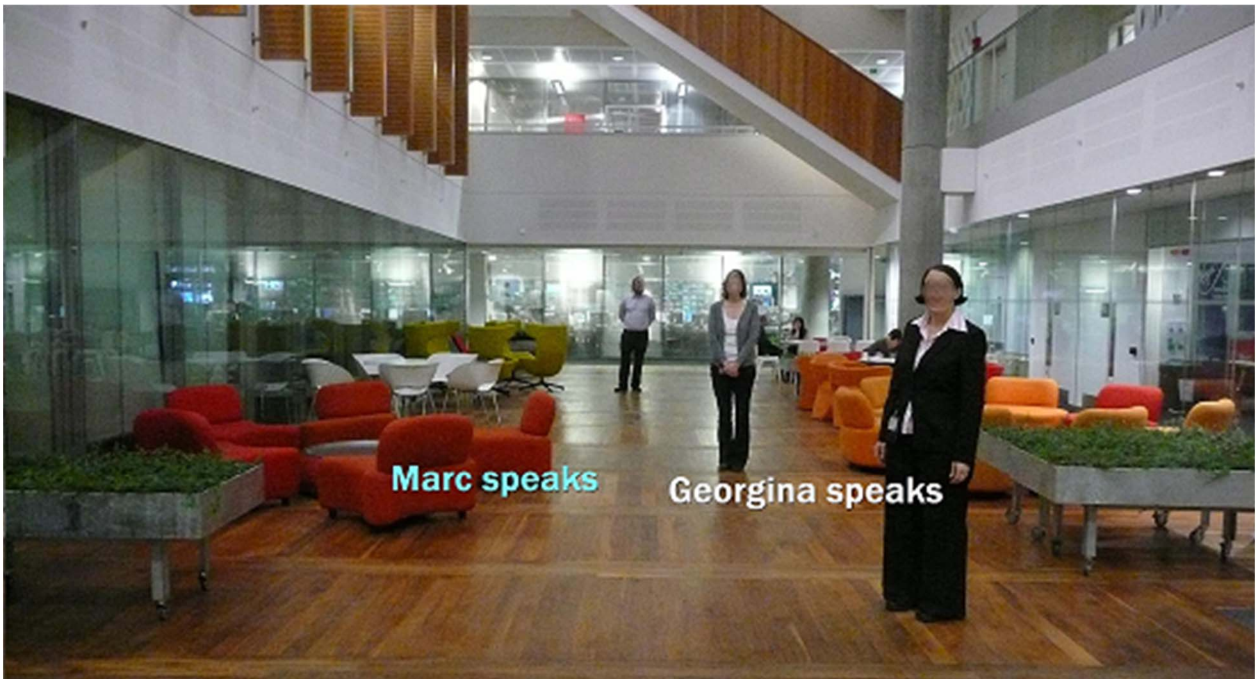


Figure 5: Different subtitles sharing a region

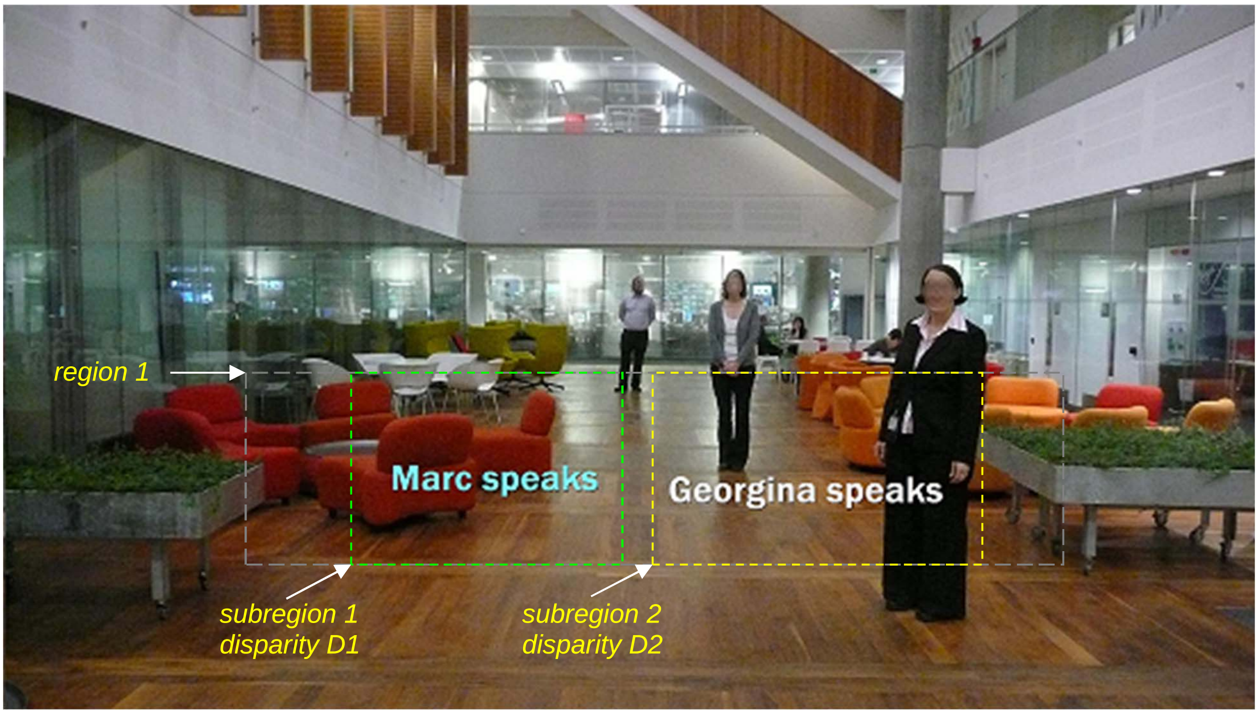


Figure 6: Different subtitles assigned to different subregions within one region

Temporal updates to disparity values may be encoded by different strategies. One simple method is to transmit successive DSSs whose signalled values are timed to the PTS of their respective PES packets. Another potentially more bit-rate efficient method uses the DSS to signal a succession of disparity updates using the disparity_shift_update_sequence mechanism defined below. Note that a mixed approach is also possible in which, for example, a DSS which includes a disparity_shift_update_sequence is followed (and possibly overruled) by a DSS with a new disparity_shift_update_sequence or by a DSS which signals a new set of disparity values timed to the PTS.

The disparity shift update sequence mechanism is illustrated in figure 7 and in annex C. A succession of near-future disparity values are transmitted together, defined at intervals which can vary, and are applied at times which can easily be calculated from the PTS and the transmitted interval parameters. Intermediate disparity values may be interpolated by the decoder as appropriate within the capabilities of the decoder (two possible interpolation approaches are indicated in figure 7 by hatched lines). Care should be taken in interpolation to avoid "overshoot" in the calculated intermediate disparity values (particularly for positive values).

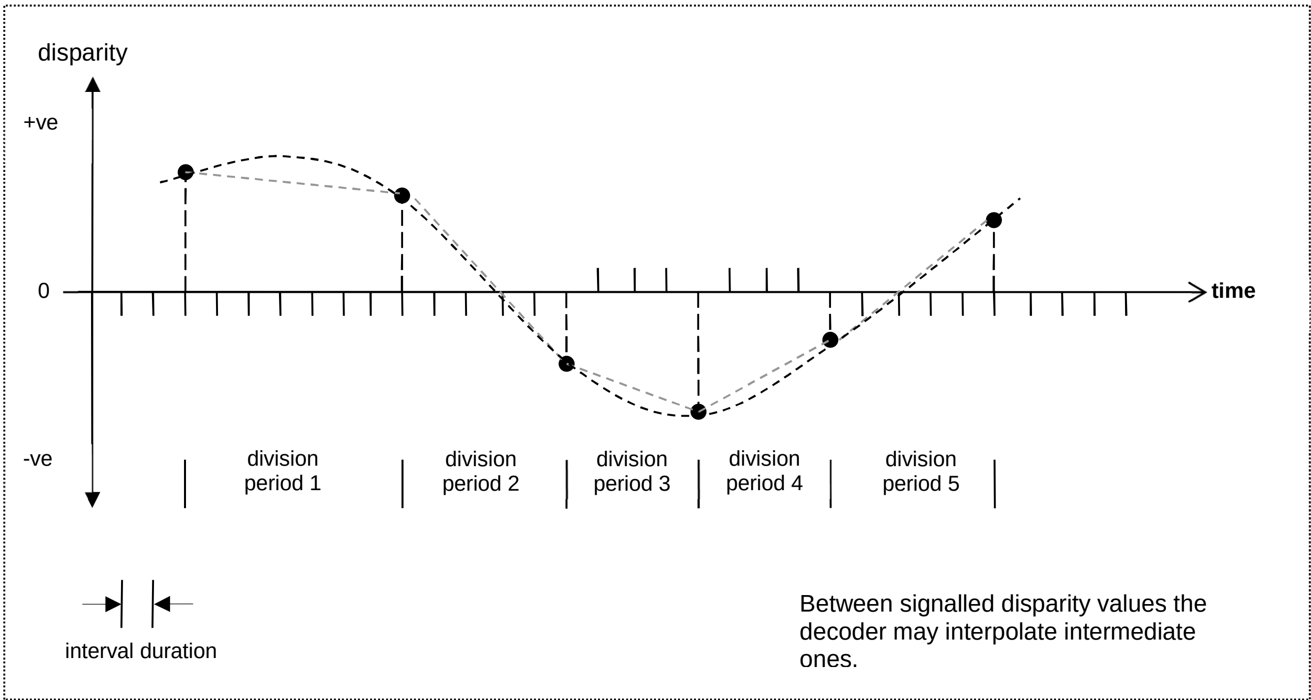


Figure 7: Disparity updates using the disparity_shift_update_sequence mechanism

Experiments have shown that some legacy 2D IRDs do not behave in a predictable and user-friendly manner when presented with subtitle streams which contain a DSS.

Broadcasters, service providers and network operators should note that services intended for 2D IRDs but derived from 3D services should therefore include subtitle streams coded in accordance with ETSI EN 300 743 (V1.3.1) [6] i.e. without a DSS. In the case of service-compatible 3D this may involve providing two subtitle streams per language carried on separate PIDs (with and without a DSS) and distinguishing the 2D and 3D versions of the service appropriately in the PSI.

The syntax of the disparity signalling segment is shown in table 29.

Table 29: Disparity signalling segment

Syntax	Size	Type
disparity_signalling_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
dss_version_number	4	uimsbf
disparity_shift_update_sequence_page_flag	1	bslbf
reserved	3	bslbf
page_default_disparity_shift	8	tcimsbf
if (disparity_shift_update_sequence_page_flag ==1) {		
disparity_shift_update_sequence()		
}		
while (processed_length<segment_length) {		
region_id	8	uimsbf
disparity_shift_update_sequence_region_flag	1	bslbf
reserved	5	uimsbf
number_of_subregions_minus_1	2	uimsbf
for (n=0; n<= number_of_subregions_minus_1; n++) {		
if (number_of_subregions_minus_1 > 0) {		
subregion_horizontal_position	16	uimsbf
subregion_width	16	uimsbf
}		
subregion_disparity_shift_integer_part	8	tcimsbf
subregion_disparity_shift_fractional_part	4	uimsbf
reserved	4	uimsbf
if (disparity_shift_update_sequence_region_flag ==1) {		
disparity_shift_update_sequence()		
}		
}		
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x15, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

dss_version_number: Indicates the version of this DSS. The version number is incremented (modulo 16) if ***any*** of the parameters for this particular DSS are modified.

disparity_shift_update_sequence_page_flag: If '1' then the disparity_shift_update_sequence immediately following is to be applied to the page_default_disparity_shift. If '0' then a disparity_shift_update_sequence for page_default_disparity_shift is not included.

page_default_disparity_shift: Specifies the default disparity value which should be applied to all regions within the page (and thus to all objects within those regions) in the event that the decoder cannot apply individual disparity values to each region. This disparity value is a signed integer and thus allows the default disparity to range between +127 and -128 pixels.

NOTE 1: Any decoder which can apply separate disparity values to a region or subregion has to apply the relevant values to any subregions signalled in the region loop.

disparity_shift_update_sequence: The syntax of this field is specified in table 30.

Table 30: disparity_shift_update_sequence

Syntax	Size	Type
disparity_shift_update_sequence() {		
disparity_shift_update_sequence_length	8	bslbf
interval_duration[23..0]	24	uimsbf
division_period_count	8	uimsbf
for (i= 0; i< division_period_count; i ++) {		
interval_count	8	uimsbf
disparity_shift_update_integer_part	8	tcimsbf
}		
}		

Semantics:

processed_length: The total number of bytes that have already been processed following the segment_length field.

region_id: Identifies the region to which the following subregion data refers. Regions which have been declared in the display set but which are not referenced in the while-loop has to adopt the page_default_disparity and its associated disparity_update_sequence where present.

disparity_shift_update_sequence_region_flag: If '1' then a disparity_shift_update_sequence is included for all subregions of this region. If '0' then a disparity_shift_update_sequence for this region is not included.

number_of_subregions_minus_1: The number of subregions minus one which apply to this region. If number_of_subregions_minus_1 = 0 then the region has only one subregion whose dimensions are the same as the region and the signalled disparity therefore applies to the whole region.

subregion_horizontal_position: Specifies the left-hand most pixel position of this subregion. This value shall always fall within the declared extent of the region of which this is a subregion and shall therefore be in the range 0..4095. Note that as with the region positional specification this horizontal position is relative to the page.

subregion_width: Specifies the horizontal width of this subregion expressed in pixels. The combination of subregion_horizontal_position and subregion_width shall always fall within the declared extent of the region to which this refers. The value of this field shall therefore be in the range 0..4095.

subregion_disparity_shift_integer_part: Specifies the integer part of the disparity shift value which should be applied to all subtitle pixel data enclosed within this subregion. This allows the disparity to range between + 127 and -128 pixels.

subregion_disparity_shift_fractional_part: Specifies the fractional part of the disparity shift value which should be applied to all subtitle pixel data enclosed within this subregion. When used as an extension of the integer part, this allows the signalled disparity shift to be defined to $^{1}/_{16}$ pixel accuracy. Note that this fractional part is unsigned (0b0001 represents $^{1}/_{16}$ pixel and 0b1111 represents $^{15}/_{16}$ pixel) and should be combined with the integer part always by adding the fractional part to the integer part. A disparity value of -0,75 is therefore signalled as [-1, 0,25] and a value of -4,5 as [-5, 0,5].

NOTE 2: Any processing (either at the encoder or the decoder) which needs to implement only integer values of disparity shift has to ensure values are rounded "towards the viewer" (i.e. that positive values of disparity are rounded down and negative values rounded up).

disparity_shift_update_sequence_length: Specifies the number of bytes contained in the disparity_shift_update_sequence which follows this field.

interval_duration: Specifies the unit of interval used to calculate the PTS for the disparity update as a 24-bit field (in 90 kHz STC increments). The value of interval_duration shall correspond to an exact multiple (≥ 1) of frame periods and its maximum value is therefore just over 186 seconds.

division_period_count: Specifies the number of unique disparity values (≥ 1) and hence the number of time intervals within the following disparity_shift_update_sequence 'for' loop.

interval_count: Specifies the multiplier used to calculate the PTS for this disparity update from the initial PTS value. The calculation for the PTS for this update is $PTS_{new} = PTS_{previous} + (interval\_duration \times interval\_count)$ where interval count ≥ 1, where PTS_{new} increases with every iteration of the loop and where the initial value of $PTS_{previous}$ is the PTS signalled in the PES header.

disparity_shift_update_integer_part: Specifies the integer part of the disparity update value which should be applied to all subtitle pixel data enclosed within this page or this subregion. This allows the disparity to excurse +127 to -128 pixels.

7.2.8 Alternative CLUT segment

The versions of the present document prior to V1.6.1 defined CLUTs exclusively in Recommendation ITU-R BT.601 [3] colour space. The alternative_CLUT_segment (ACS) permits a CLUT to be defined in other colour systems. The syntax of the ACS is shown in table 31.

For the purpose of optimal backwards compatibility of subtitle services and existing decoders, when a subtitle service makes use of the alternative_CLUT_segment (ACS), it shall also provide the legacy capability of rendering in the ITU-R BT.601 [3] colour space, by the provision of a CDS within the same CLUT family (with the same CLUT_id) that contains the same number of entries as the ACS, so that IRDs that do not support the ACS can perform their own conversion from the Recommendation ITU-R BT.601 [3] colours for the rendering of the subtitles with non-Recommendation ITU-R BT.601 [3] video content.

The ACS permits a CLUT with up to 256 colours. This allows a sufficient number of colours to be used in order to achieve high quality anti-aliasing. This mitigates the effects of spatial upscaling, especially with UHDTV services.

Table 31: Alternative CLUT segment

Syntax	Size	Type
alternative_CLUT_segment() {		
sync_byte	8	bslbf
segment_type	8	bslbf
page_id	16	bslbf
segment_length	16	uimsbf
CLUT_id	8	bslbf
CLUT_version_number	4	uimsbf
reserved_zero_future_use	4	bslbf
CLUT_parameters()	16	bslbf
while (processed_length < segment_length) {		
If (output_bit_depth == 0) {		
luma-value	8	uimsbf
chroma1-value	8	uimsbf
chroma2-value	8	uimsbf
T-value	8	uimsbf
}		
If (output_bit_depth == 1) {		
luma-value	10	uimsbf
chroma1-value	10	uimsbf
chroma2-value	10	uimsbf
T-value	10	uimsbf
}		
}		
}		

Semantics:

sync_byte: This field shall contain the value '0000 1111'.

segment_type: This field shall contain the value 0x16, as listed in table 7.

page_id: The page_id identifies the subtitle service of the data contained in this subtitling_segment. Segments with a page_id value signalled in the subtitling descriptor as the composition page id, carry subtitling data specific for one subtitle service. Accordingly, segments with the page_id signalled in the subtitling descriptor as the ancillary page id, carry data that may be shared by multiple subtitle services.

segment_length: This field shall indicate the number of bytes contained in the segment following the segment_length field.

CLUT_id: This field identifies within a page the CLUT family whose data is contained in this alternative_CLUT_segment field. Its value shall be the same as for the CLUT_id contained in the CDS of the same subtitle service.

CLUT_version_number: Indicates the version of this segment data. When any of the contents of this segment change this version number is incremented (modulo 16).

reserved_zero_future_use: These bits are reserved for future use. They shall be set to the value 0x0.

CLUT_parameters: This 16-bit field has the syntax as shown in table 32.

Table 32: CLUT parameters

Syntax	Size	Type
CLUT_parameters() {		
CLUT_entry_max_number	2	bslbf
colour_component_type	2	bslbf
output_bit_depth	3	bslbf
reserved_zero_future_use	1	bslbf
dynamic_range_and_colour_gamut	8	bslbf
}		

Semantics:

CLUT_entry_max_number: This two-bit field shall indicate the maximum number of CLUT entries. A value of '0' corresponds to a maximum number of 256 entries. All other values are reserved. Any number of CLUT entries can be provided, up to the maximum number.

colour_component_type: This two-bit field shall indicate the type of colour coding used in the chroma1-value and chroma2-value fields. A value of '0' corresponds to colour coding type YCbCr, whereby chroma1-value is Cb and chroma2-value is Cr. All other values are reserved.

output_bit_depth: This three-bit field shall indicate the bit-depth of the output of each component, as shown in table 33. If the graphics plane of the IRD has a bit-depth different from the output_bit_depth setting, then the IRD shall perform the appropriate conversion for each component value of the CLUT.

Table 33: Output bit-depth coding

Value	Output bit-depth
0x0	8
0x1	10
0x2 - 0x7	Reserved

reserved_zero_future_use: This bit is reserved for future use. It shall be set to the value 0.

dynamic_range_and_colour_gamut: This eight-bit field shall be coded according to one of the entries in table 34.

Table 34: Dynamic range and colour gamut coding

Value	Dynamic range and colour gamut
0x00	SDR; ITU-R BT.709 [10]
0x01	SDR; ITU-R BT.2020-2 [11]
0x02	HDR; ITU-R BT.2100-1 [12] PQ
0x03	HDR; ITU-R BT.2100-1 [12] HLG
0x04 - 0xFF	Reserved

luma-value: This field indicates the luma output value of the CLUT entry.

chroma1-value: This field indicates the first chroma output value of the CLUT entry.

chroma2-value: This field indicates the second chroma output value of the CLUT entry.

T-value: This field indicates the transparency value of the CLUT entry.

In contrast to the syntax of the CDS, the ACS syntax does not contain an explicit parameter for CLUT_entry_id. This is due to there being no preset default colours for the colour systems covered by the ACS, so that the CLUT containing the complete set of colours used in the subtitle service shall always be provided when a subtitle service uses the ACS. The IRD shall assume CLUT entry id numbers for each CLUT entry in the order of appearance of the set of luma, chroma1, chroma2 and T values in the CLUT entry data block of the ACS, starting from entry id '0'.

The IRD shall ignore an ACS if any field within its CLUT_parameters() structure is set to an unsupported or reserved value.

7.3 Interoperability points

The present clause specifies four interoperability points for subtitle services and decoders. These are based on the four TV service classes SDTV, HDTV, 3DTV and UHDTV, whereby certain exceptions and combinations are possible, as also specified in the present clause. Table 35 collects the various compliance requirements for the four interoperability points for subtitle decoders, for all relevant aspects of compliance that are within the scope of the present document.

The aspect of ETSI EN 300 743 version compliance provides informative guidance on which version of the present document applies to the corresponding profile.

Within the category "subtitle stream composition" only the segment types are listed that are not mandatory to be supported for all subtitle decoders. The segment types region composition segment, page composition segment, CLUT definition segment, object data segment and end of display set segment shall be supported by all subtitle decoders. For the object data segment there are different interoperability requirements based on the object coding method. Object coding method '0', coding of pixels, shall be supported by all subtitle decoders.

Table 35: Subtitle decoder interoperability points

Aspect of compliance	Feature	Subtitle decoder interoperability point			
		IRD with "SDTV" subtitling support	IRD with "HDTV" subtitling support	IRD with "3DTV" subtitling support	IRD with "UHDTV" subtitling support
ETSI EN 300 743 version compliance	N/A	1.1.1, 1.2.1	1.3.1	1.4.1, 1.5.1	1.6.1
Service Information	Subtitling type coding (see ETSI EN 300 468 [2])	0x10-0x13, 0x20-0x23	0x14, 0x24	0x15, 0x25	0x16, 0x26
Subtitle stream composition	Display definition segment (DDS) (specified in clause 7.2.1)	Not applicable	Mandatory	Mandatory	Mandatory
	Disparity signalling segment (DSS) (specified in clause 7.2.7)	Not applicable	Not applicable	Mandatory	Conditional Mandatory (see note)
	Alternative CLUT segment (ACS) (specified in clause 7.2.8)	Not applicable	Not applicable	Not applicable	Optional
Object data segment (ODS)	Interlaced coding of pixels (method '0') (specified in clauses 7.2.5.1 and 7.2.5.2)	Mandatory	Mandatory	Mandatory	Mandatory
	Coding as a string of characters (method '1')	Undefined	Undefined	Undefined	Undefined
	Progressive coding of pixels (method '2') (specified in clause 7.2.5.3)	Not applicable	Not applicable	Not applicable	Mandatory
Forward compatibility	As specified in clause 7.2.0.2	Recommended	Recommended	Recommended	Mandatory
NOTE: The DSS shall be supported by IRDs that support 3DTV services. Other IRDs need not support the DSS.					

Subtitle service interoperability points corresponding to the four subtitle decoder interoperability points are derived by the usage of only those subtitling features that shall or may be supported by the corresponding subtitle decoder interoperability point.

8 Requirements for the subtitling data

8.0 General

Unless stated otherwise, all requirements apply at any particular point in time but they do not relate to situations at different points in time. In this clause the following terminology is used. If a segment is signalled by the composition page id value, then the segment is said to be "in" the composition page and the composition page is said to "contain" that segment. Similarly, a segment signalled by the ancillary page id value is said to be "in" the ancillary page and the ancillary page is said to "contain" such segment.

The page id value of a segment containing data for a subtitle service shall be equal either to the value of the composition_page_id or the ancillary_page_id provided in the subtitling descriptor. Page compositions are not shared by multiple subtitle services; consequently, the page id of each page composition segment shall be equal to the composition_page_id value.

Within a subtitle stream, a page id value is assigned to each segment. Segments can either contain data specific for one subtitle service, or data that is to be shared by more than one subtitle service. The data for a subtitle service shall be carried in segments identified by at most two different page id values:

- one page id value signalling segments with data specific for that subtitle service; the use of this type of data is mandatory;
- one page id value signalling segments with data that may be shared by multiple subtitle services; the use of this type of data is optional.

All segments signalled by the composition page id value shall be delivered before any segment signalled by the ancillary page id value. The ancillary page id value shall not signal page composition segments and region composition segments.

8.1 Scope of Identifiers

All identifiers (region_id, CLUT_id, object_id) shall be unique within a page.

8.2 Scope of dependencies

8.2.1 Composition page

A segment in the composition page may reference segments in that composition page as well as segments in the ancillary page.

All segments signalled by the composition page id value shall be delivered before any segment signalled by the ancillary page id value.

8.2.2 Ancillary page

The ancillary page may contain only CLUT definition segments, alternative CLUT segments, and object data segments. Neither page composition segments, nor region composition segments shall be carried in the ancillary page. Segments in an ancillary page can be referenced by segments in any (composition) page.

Segments signalled by the ancillary page id value shall be delivered after all segments signalled by the composition page id value.

NOTE: From clauses 8.2.1 and 8.2.2 it follows that segments in a composition page are able to be referenced only by segments in the same composition page.

8.3 Order of delivery

The PTS field in successive subtitling PES packets shall either remain the same or proceed monotonically. Thus subtitling PES packets shall be delivered in their presentation time-order.

The PTSs of subsequent display sets shall differ by at least one video frame period. Discontinuities in the PTS sequence may occur if there are discontinuities in the PCR time base.

8.4 Positioning of regions and objects

8.4.1 Regions

A region monopolizes the scan lines on which it is shown; no two regions can be presented horizontally next to each other.

8.4.2 Objects sharing a PTS

Objects that are referenced by the same PTS (i.e. they are part of the same display set) shall not overlap on the screen.

8.4.3 Objects added to a region

If an object is added to a region, the new pixel data will overwrite the present information in the region. Thus a new object may (partly) cover old objects. The programme provider shall take care that the new pixel data overwrites only information that needs to be replaced, but also that it overwrites all information on the screen that is not to be preserved.

NOTE: A pixel is either defined by an "old" object or by the background colour or by the "new" object; if a pixel is overwritten none of its previous definition is retained.

9 Translation to colour components

9.0 General

The present clause applies to subtitle services that are authored in accordance with Recommendation ITU-BT.601 [3] and thus make use of only the default CLUT(s) and/or the CLUT definition segment (CDS) to determine their appearance. Subtitle services that are authored for other colour and dynamic range systems, thus making use of the alternative CLUT segment (ACS) and 8-bit default-CLUT and/or CDS, shall not apply the CLUT translation processes specified in the present clause.

Translation processes need to be applied when CLUT reduction is performed by decoders that do not support either the 4-bit CLUT and/or the 8-bit CLUT options for the coding of subtitle services, in order to be able to nevertheless display subtitles coded with 4- and/or 8-bit CLUTs, albeit in a cruder form. Subtitle services can indicate that these CLUT reduction techniques shall not be applied by specifying the minimum compatibility in the region composition segment (see clause 7.2.3).

The subtitling system directly supports IRDs that can present four colours, sixteen colours and 256 colours, respectively. The requirements related to translation for the three cases of IRD are specified as follows:

- 4 colour IRDs. Pixel codes that use a 2-bit CLUT can be decoded into Y, Cr, Cb and T directly; pixel codes that use a 4-bit or 8-bit CLUT can be decoded also, but only if the region allows for decoding on a 2-bit CLUT. If such decoding is allowed, reduction schemes are provided for translating the original 16 or 256 colours to the available 4 colours, in clauses 9.1 and 9.2 respectively.

- 16 colour IRDs. Pixel codes that use a 2-bit or 4-bit CLUT can be decoded into Y, Cr, Cb and T directly; pixel codes that use an 8-bit CLUT can be decoded if the region allows for decoding on a 4-bit CLUT. If such decoding is allowed, a reduction scheme is provided in clauses 9.3 for translating the original 256 colours to the available 16 colours. When pixel codes use a 4-bit CLUT, it is possible to switch to a 2-bit coding scheme within certain areas where at most 4 out of the 16 available colours are used. This requires a map table specifying which 4 CLUT entries are addressed with the 2-bit codes.
- 256 colour IRDs. All pixel codes can be decoded into Y, Cr, Cb and T directly, irrespective whether they use a 2-bit or 4-bit or an 8-bit CLUT. When a pixel code uses a 4-bit or an 8-bit CLUT, it is possible to switch to a 2-bit or a 4-bit coding scheme within a certain area where at most 4 or 16 out of the 256 available colours are used. This requires a map table specifying which 4 or 16 CLUT entries are addressed with the 2-bit or 4-bit codes, respectively.

The IRD shall translate a pixel's pseudo-colours into Y, Cr, Cb and T components according to the model depicted in figure 8.

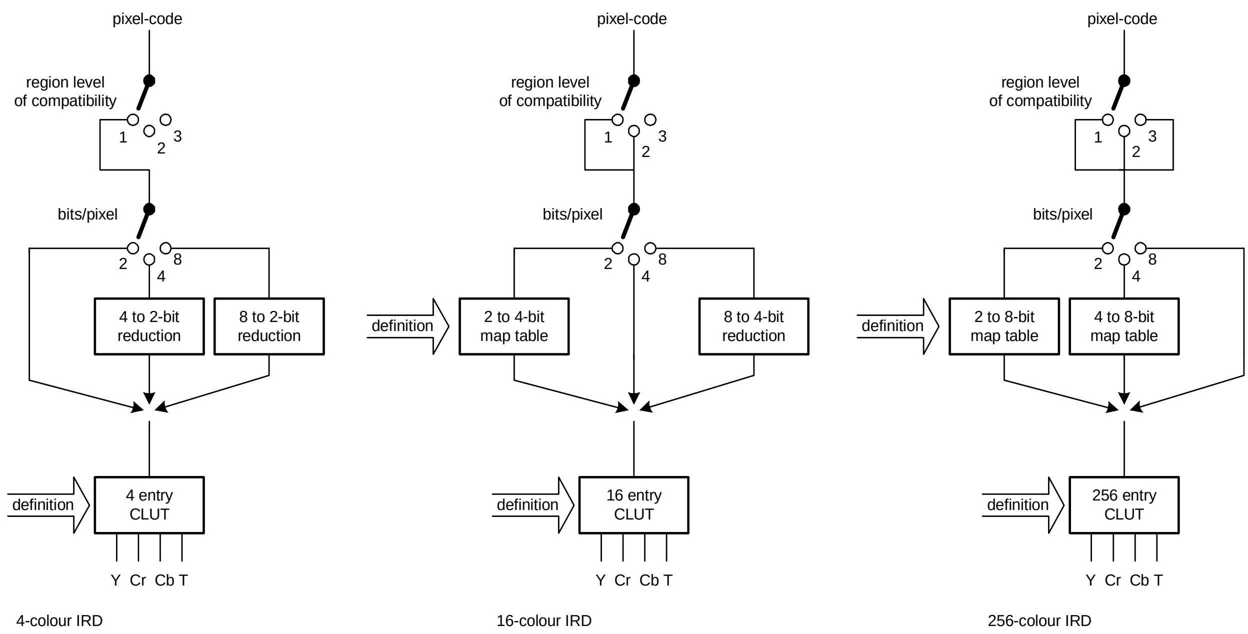


Figure 8: IRD subtitle colour translation model

9.1 4- to 2-bit reduction

Let the input value be represented by a 4-bit field, the individual bits of which are called b_{i1}, b_{i2}, b_{i3} and b_{i4} where b_{i1} is received first and b_{i4} is received last. Let the output value be represented by a 2-bit field b_{o1}, b_{o2}.

The relation between output and input bits is:

$$b_{o1} = b_{i1}$$

$$b_{o2} = b_{i2} \mid b_{i3} \mid b_{i4}$$

9.2 8- to 2-bit reduction

Let the input value be represented by an 8-bit field, the individual bits of which are called bi1, bi2, bi3, bi4, bi5, bi6, bi7 and bi8 where bi1 is received first and bi8 is received last. Let the output value be represented by a 2-bit field bo1, bo2.

The relation between output and input bits is:

$b_{o1} = b_{i1}$

$b_{o2} = b_{i2} \mid b_{i3} \mid b_{i4}$

9.3 8- to 4-bit reduction

Let the input value be represented by a 8-bit field, the individual bits of which are called bi1, bi2, bi3, bi4, bi5, bi6, bi7 and bi8 where bi1 is received first and bi8 is received last. Let the output value be represented by a 4-bit field bo1 to bo4.

The relation between output and input bits is:

$b_{o1} = b_{i1}$ $b_{o2} = b_{i2}$

$b_{o3} = b_{i3}$ $b_{o4} = b_{i4}$

10 Default CLUTs and map-tables contents

10.0 General

This clause specifies the default contents of the CLUTs and map-tables for every CLUT family. Every entry for every CLUT can be redefined in a CLUT_definition_segment and every map-table can be redefined in an object_data_segment, but before such redefinitions the contents of CLUTs and map-tables shall correspond to the values specified here.

CLUTs may be redefined partially. Entries that have not been redefined shall retain their default contents.

10.1 256-entry CLUT default contents

The CLUT is divided in six sections: 64 colours of reduced intensity 0 to 50 %, 56 colours of higher intensity 0 % to 100 %, 7 colours with 75 % transparency, 1 "colour" with 100 % transparency, 64 colours with 50 % transparency and 64 light colours (50 % white + colour 0 % to 50 %).

Let the CLUT-entry number be represented by an 8-bit field, the individual bits of which are called b_1, b_2, b_3, b_4, b_5, b_6, b_7 and b_8 where b_1 is received first and b_8 is received last. The value in a bit is regarded as unsigned integer that can take the values zero and one.

The resulting colours are described here in terms of Red, Green and Blue contributions, as shown in table 36. To find the CLUT contents in terms of Y, Cr and Cb components, see Recommendation ITU-R BT.601 [3].

Table 36: 256-entry CLUT default contents

if b_1 == '0' && b_5 == '0' {
if b_2 == '0' && b_3 == '0' && b_4 == '0' {
if b_6 == '0' && b_7 == '0' && b_8 == '0'
T = 100 %
else {
R = 100 % $\times$ b_8
G = 100 % $\times$ b_7
B = 100 % $\times$ b_6
T = 75 %
}
}
else {
R = 33,3 % $\times$ b_8 + 66,7 % $\times$ b_4
G = 33,3 % $\times$ b_7 + 66,7 % $\times$ b_3
B = 33,3 % $\times$ b_6 + 66,7 % $\times$ b_2
T = 0 %
}
}
if b_1 == '0' && b_5 == '1' {
R = 33,3 % $\times$ b_8 + 66,7 % $\times$ b_4
G = 33,3 % $\times$ b_7 + 66,7 % $\times$ b_3
B = 33,3 % $\times$ b_6 + 66,7 % $\times$ b_2
T = 50 %
}
if b_1 == '1' && b_5 == '0' {
R = 16,7 % $\times$ b_8 + 33,3 % $\times$ b_4 + 50 %
G = 16,7 % $\times$ b_7 + 33,3 % $\times$ b_3 + 50 %
B = 16,7 % $\times$ b_6 + 33,3 % $\times$ b_2 + 50 %
T = 0 %
}
if b_1 == '1' && b_5 == '1' {
R = 16,7 % $\times$ b_8 + 33,3 % $\times$ b_4
G = 16,7 % $\times$ b_7 + 33,3 % $\times$ b_3
B = 16,7 % $\times$ b_6 + 33,3 % $\times$ b_2
T = 0 %
}

10.2 16-entry CLUT default contents

Let the CLUT-entry number be represented by a 4-bit field, the individual bits of which are called b_1, b_2, b_3 and b_4 where b_1 is received first and b_4 is received last. The value in a bit is regarded as unsigned integer that can take the values zero and one.

The resulting colours are described here in terms of Red, Green and Blue contributions, as shown in table 37. To find the CLUT contents in terms of Y, Cr and Cb components, please see Recommendation ITU-R BT.601 [3].

Table 37: 16-entry CLUT default contents

if b_1 == '0' {
if b_2 == '0' && b_3 == '0' && b_4 == '0' {
T = 100 %
}
else {
R = 100 % $\times$ b_4
G = 100 % $\times$ b_3
B = 100 % $\times$ b_2
T = 0 %
}
}
if b_1 == '1' {
R = 50 % $\times$ b_4
G = 50 % $\times$ b_3
B = 50 % $\times$ b_2
T = 0 %
}

10.3 4-entry CLUT default contents

Let the CLUT-entry number be represented by a 2-bit field, the individual bits of which are called b_1 and b_2 where b_1 is received first and b_2 is received last.

The resulting colours are described here in terms of Red, Green and Blue contributions, as shown in table 38. To find the CLUT contents in terms of Y, Cr and Cb components, please see Recommendation ITU-R BT.601 [3].

Table 38: 4-entry CLUT default contents

if b_1 == '0' && b_2 == '0' {
T = 100 %
}
if b_1 == '0' && b_2 == '1' {
R = G = B = 100 %
T = 0 %
}
if b_1 == '1' && b_2 == '0' {
R = G = B = 0 %
T = 0 %
}
if b_1 == '1' && b_2 == '1' {
R = G = B = 50 %
T = 0 %
}

10.4 2_to_4-bit_map-table default contents

The 2_to_4-bit_map-table default contents are specified in table 39.

Table 39: 2_to_4-bit_map-table default contents

Input value	Output value
00	0000
01	0111
10	1000
11	1111

Input and output values are listed with their first bit left.

10.5 2_to_8-bit_map-table default contents

The 2_to_8-bit_map-table default contents are specified in table 40.

Table 40: 2_to_8-bit_map-table default contents

Input value	Output value
00	0000 0000
01	0111 0111
10	1000 1000
11	1111 1111

Input and output values are listed with their first bit left.

10.6 4_to_8-bit_map-table default contents

The 4_to_8-bit_map-table default contents are specified in table 41.

Table 41: 4_to_8-bit_map-table default contents

Input value	Output value
0000	0000 0000
0001	0001 0001
0010	0010 0010
0011	0011 0011
0100	0100 0100
0101	0101 0101
0110	0110 0110
0111	0111 0111
1000	1000 1000
1001	1001 1001
1010	1010 1010
1011	1011 1011
1100	1100 1100
1101	1101 1101
1110	1110 1110
1111	1111 1111

Input and output values are listed with their first bit left.

11 Structure of the pixel code strings (informative)

The structure of the 2-bit/pixel_code_string is shown in table 42.

Table 42: 2-bit/pixel_code_string()

Value	Meaning
01	one pixel in colour 1
10	one pixel in colour 2
11	one pixel in colour 3
00 01	one pixel in colour 0
00 00 01	two pixels in colour 0
00 1L LL CC	L pixels (3..10) in colour C
00 00 10 LL LL CC	L pixels (12..27) in colour C
00 00 11 LL LL LL LL CC	L pixels (29..284) in colour C
00 00 00	end of 2-bit/pixel_code_string
NOTE: Runs of 11 pixels and 28 pixels can be coded as one pixel plus a run of 10 pixels and 27 pixels, respectively.	

The structure of the 4-bit/pixel_code_string is shown in table 43.

Table 43: 4-bit/pixel_code_string()

Value	Meaning
0001 To 1111	one pixel in colour 1 to one pixel in colour 15
0000 1100	one pixel in colour 0
0000 1101	two pixels in colour 0
0000 0LLL	L pixels (3..9) in colour 0 (L>0)
0000 10LL CCCC	L pixels (4..7) in colour C
0000 1110 LLLL CCCC	L pixels (9..24) in colour C
0000 1111 LLLL LLLL CCCC	L pixels (25..280) in colour C
0000 0000	end of 4-bit/pixel_code_string
NOTE: Runs of 8 pixels in a colour not equal to '0' can be coded as one pixel plus a run of 7 pixels.	

The structure of the 8-bit/pixel_code_string is shown in table 44.

Table 44: 8-bit/pixel_code_string()

Value	Meaning
00000001 To 11111111	one pixel in colour 1 to one pixel in colour 255
00000000 0LLLLLLL	L pixels (1-127) in colour 0 (L > 0)
00000000 1LLLLLLL CCCCCCCC	L pixels (3-127) in colour C (L > 2)
00000000 00000000	end of 8-bit/pixel_code_string

12 Subtitle rendering issues

12.1 Introduction

This clause provides guidelines around the rendering of subtitles. Attention is needed to this aspect due to DVB specifications having evolved from the original SDTV services to HDTV, 3DTV, and UHDTV services, since the first edition of the present document. With these enhancements come extended screen resolutions and enhanced video colour systems, which all have some impact on the rendering of subtitles.

The following clauses deal with each particular aspect.

12.2 Spatial scaling of subtitles

HDTV and UHDTV decoders that offer a means of scaling or positioning the subtitles under user control (e.g. to make them larger or smaller) can use the information conveyed in the display definition segment (DDS) to determine safe strategies for zooming and/or positioning that will ensure that windowed subtitles can remain visible and readable.

It is generally recommended that subtitle graphics are anti-aliased and produced at the native resolution of the expected display. If the graphics are not created at the native display resolution, they need to be scaled and if this scaling is not done carefully the quality of subtitles can be degraded significantly, resulting in reduced readability.

The DDS provides an optional display window feature. When the display window feature is used it allows smaller, more efficient graphics to be produced, with the trade-off of restricting the subtitles to only part of the display area. Scaling is not required if the display_width and display_height fields in the DDS match the display resolution.

If the display window feature is not used and the display_width and display_height in the DDS do not match the display resolution, then the subtitle graphics should be scaled by the IRD with appropriate filtering before being overlaid on the video.

For a UHDTV service, the size of subtitles graphics is recommended to not exceed 1 920 by 1 080 pixels. If the display window is not used, an IRD with the maximum UHDTV display resolution (3 840 by 2 160 pixels) can apply a resolution upscaling by a factor of two in both horizontal and vertical directions. If a display window is used, and the display_width and display_height match the display resolution, the subtitle graphics can be rendered directly onto the part of the UHDTV display specified in the DDS.

Apart from such conversions from SDTV to HDTV resolution, or from HDTV to UHDTV resolution, more arbitrary scaling operations should be avoided wherever possible for subtitles that have been anti-aliased for their original graphical resolution. Any scaling applied to such subtitles could degrade them significantly and thereby impact their readability.

12.3 Rendering subtitles over video with a different colour system

This clause concerns the rendering of subtitles over video content that uses a colour system other than Recommendation ITU-R BT.601 [3], used in SDTV systems.

HDTV systems use Recommendation ITU-R BT.709 [10], but that distinction was not taken into account when the present document was revised to include HDTV-resolution subtitles (V1.3.1). In common practice IRDs use Recommendation ITU-R BT.709 [10] for rendering both SDTV and HDTV video services, due to the minimal difference in the results on the screen.

With the advent of UHDTV and HDR video, however, care is needed with the rendering of subtitles so that their readability and intended appearance is maintained when displayed on top of UHDTV and HDR video. In principle there is the risk with HDR video that the readability of overlaid subtitles might be impacted when the video scene in the background contains high luminance levels outside the range of that available to the overlaid subtitles. In general, however, HDR video scenes will not include high luminance levels over large areas of the scene over long periods of time, rather high luminance levels will be localized on the screen, e.g. in the form of specular highlights. Assuming the IRD performs an adequate conversion of the CLUT colours, in Recommendation ITU-R BT.601 [3] colour space, contained in the default CLUTs and CLUT definition segment (CDS) for rendering, it is not expected that HDR video will impact the readability and intended appearance of subtitles in practice. However, in V1.6.1 of the present document, the facility was introduced to enable service and content providers to provide explicitly the CLUT for video systems other than Recommendation ITU-R BT.601 [3]. This is done by using the alternative CLUT segment (ACS). In this way a more deterministic representation of the subtitles will be given on IRDs that support the ACS, giving more control over the artistic intent with subtitles over HDR video.

For subtitle services that do not provide the ACS for the non-Recommendation ITU-R BT.601 [3] target video colour and dynamic range system, or for IRDs that do not support the ACS, the IRD is recommended to convert the colours used (from the default CLUT and any entries changed via the CDS) as follows: in line with the interim guideline regarding the mapping of SDR video content into the HLG10 container according to Recommendation ITU-R BT.2100-1 [12], the luminance of subtitles should be mapped such that SDR 90 % of the narrow range signal is mapped to HLG10 75 % of the narrow range signal.

In general the approach to converting subtitle colours from the scene-referred standard dynamic range video systems, i.e. Recommendation ITU-R BT.601 [3], Recommendation ITU-R BT.709 [10] and Recommendation ITU-R BT.2020-2 [11] (all of which are SDR systems), in the IRD should take into account that the production choice of subtitle colours, in terms of both luminance and chrominance, has been performed in a dim reference viewing environment in front of an SDR reference monitor calibrated to 100 cd/m^2 according to Recommendation ITU-R BT.1886 [13].

Annex A (informative): How the DVB subtitling system works

A.0 Introduction

There are several possible ways to make the DVB subtitling system work. Aspects of several, incompatible approaches are described in the normative part of the present document.

Epoch boundaries (where page_state = "mode change") provide convenient service acquisition points. Short epochs will lead to quick service acquisition times. However, it is difficult to maintain smooth decoding across epoch boundaries and this is also likely to require more data to be broadcast. This is very similar to the issue of short GOP in MPEG video.

The main issue is to allow the decoder to keep the last valid subtitle on the display until there is a new subtitle to replace it. This requires both subtitles being in the display memory at the same time. If each display takes up less than half the pixel buffer memory it should be possible for the decoder to switch between displays smoothly. However, there is a danger of the memory becoming fragmented over several epochs. If the decoder has to perform garbage collection it may be difficult to maintain its performance.

In practice the memory plan is likely to be identical for long periods. So, it would be useful if the broadcast data could differentiate new memory plans (justifying complete destruction of state) from repeat broadcasts of old memory plans (to provide service acquisition points).

It is expected that the screen may go blank for a short period when a new memory plan is issued. At service acquisition points practical decoders will continue decoding (building on the content of the regions that they have already decoded). Decoders newly acquiring the service are recommended to erase the regions to the defined background colour and then start decoding objects into them. Clearly after acquisition the display may be incomplete until sufficient objects have been received. It is up to the broadcaster to decide how rapidly to refresh the display.

A.1 Data hierarchy and terminology

The text of clause A.1, as present in earlier releases of the DVB Subtitling Specification, has been moved into the corresponding informative clause 4.7 of the present document.

A.2 Temporal hierarchy and terminology

The text of clause A.2, as present in earlier releases of the DVB Subtitling Specification, has been moved into the corresponding informative clause 4.8 of the present document.

A.3 Decoder temporal model

The text of clause A.3, as present in earlier releases of the DVB Subtitling Specification, has been integrated into normative clause 5.1 of the present document.

A.4 Decoder display technology model

A.4.1 Region based with indexed colours

The DVB subtitling system is a region based, indexed colour, graphics system. This well matches the region-based on-screen displays being implemented at the time of writing. Such systems allow displays to be constructed using small amounts of memory. They also permit a number of apparently rapid graphical effects to be performed.

The display system can be implemented in other ways.

However, some effects that are simple when implemented in region based/indexed colour systems, may cause much greater demands when implemented in other ways. For example, in a region based system regions can be repositioned, or made visible/invisible with very little processing burden. In a simple bit mapped system such operations will require the pixel data to be moved within the display store or between the display store and some non-displayed storage. Similarly, in indexed colour systems certain effects can be implemented by redefining the contents of the CLUT associated with a particular region. In a system where there is one global CLUT for the complete display, or where pixels are not indexed before output (i.e. true colour) a CLUT redefinition may require the region to be redrawn.

The specification makes demands which are assumed to be reasonable in a region based, indexed colour, graphics system. Implementers are free to implement the graphics system in other ways. However, it is their responsibility to compensate for the implications of using an architecture that is different from that envisaged in the subtitle decoder model.

A.4.2 Colour quantization

At the time of design it was felt that some applications of the subtitling system would benefit from a 256 colour (i.e. 8-bit pixel) display system. However, it was understood that initially many decoders would have only 4- or 16-colour graphics systems.

Accordingly, the DVB subtitling system allows 256 colour graphics to be broadcast but then provides a model by which the whole spectrum of 256 colours can be quantized to 16 or 4 colours. The intention is to offer broadcasters and equipment manufacturers both a route and an incentive to move to 256 colour systems while allowing introduction of subtitling services at a time when many systems will not be able to implement 256 colours.

A byproduct of this colour quantization model is that it may be possible to implement systems with less pixel buffer memory than the 60 kbytes specified in the decoder model while still giving useful functionality. The 60 kbytes pixel buffer memory can be partitioned into any mix of 8, 4 and 2 bit per pixel regions, covering between 60 k and 240 k pixels. If memory in the decoder is very limited it may be possible to implement regions using a reduced pixel depth. For example, a region could be implemented using 2- or 4-bit pixel depth where 8 bits is the intended pixel depth.

Quantizing the colour depth may also allow the subtitling system to work with slower processors as the number of bit operations may decrease with the shallower pixel depth.

Taking full advantage of these techniques will depend on certain implementation features in the decoder. For example, it may require that the pixel depth can be set per region.

There are also broadcaster requirements to make broadcast data suitable for this approach. For example, if the broadcaster sets the region_level_of_compatability equal to the region_depth the decoder is forbidden to quantize the pixel depth. Also, if the broadcaster uses a very large number of 2-bit pixels the decoder has no opportunity to quantize colours.

A.5 Examples of the subtitling system in operation

A.5.1 Double buffering

A.5.1.0 General

Regions can be operated on while they are not visible. Also they can be made visible or invisible by modifying the region list in the page composition segment or by modifying the CLUT. These features allow a number of effects as follows.

A.5.1.1 Instant graphics

At the start of an epoch a display is defined as using 3 regions [A, B, C]. Region A is allocated to hold a station logo and so will be present in all PCS. Its content is delivered in the first display set and thereafter periodically repeated to refresh it.

Throughout the epoch PCSs will alternate between having regions A and B or A and C in their region list. When the currently active page instance uses regions A and B the decoder will be decoding the next display which will use regions A and C. As at this time region C is not visible the viewer will not see the graphics being rendered into region C. When the new display becomes valid the decoder (assuming that it has a linked list, region based, graphics system) need only modify its display list to switch from a display of regions A and B to one using regions A and C.

This approach allows the display presented to the viewer to change crisply. However, more object data may need to be broadcast (e.g. to update B to be like C).

Figures A.1 to A.5 illustrate this. The right hand side of each picture shows the display presented to the viewer. Data is always rendered into regions that are not in the display list of the currently active PCS. So, the viewer never sees data being decoded into the display.

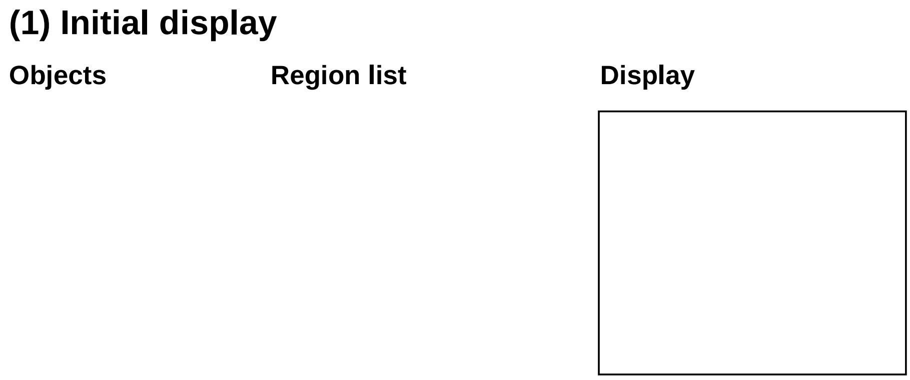


Figure A.1: Initial display (empty)

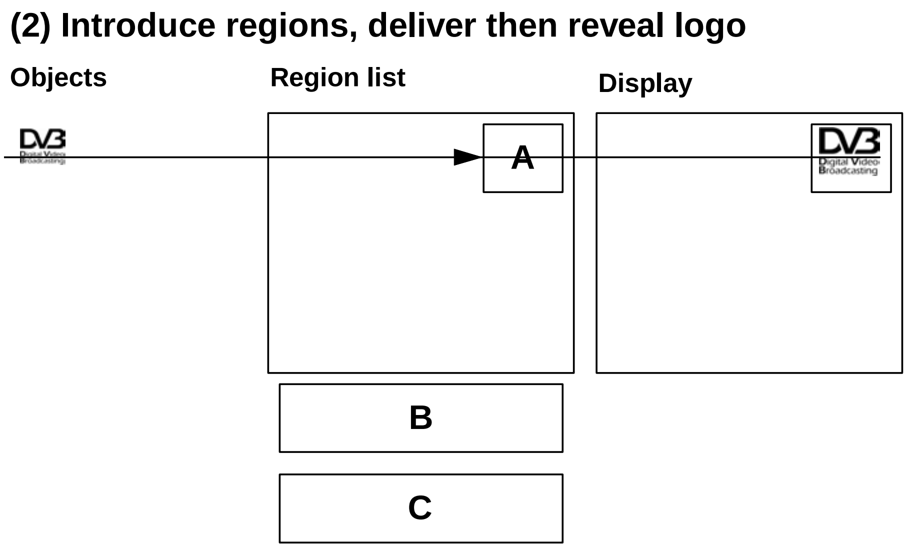


Figure A.2: Introduce regions, deliver then reveal logo

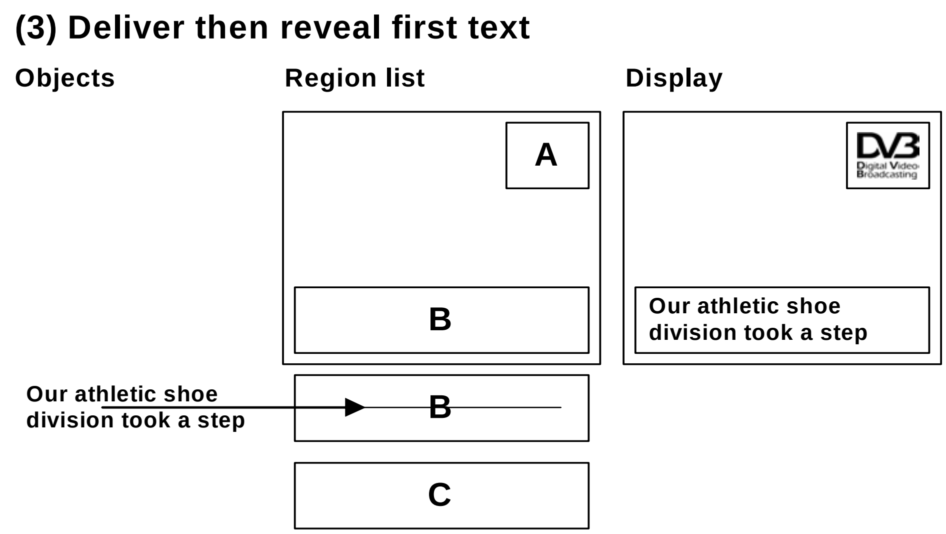


Figure A.3: Deliver then reveal first text

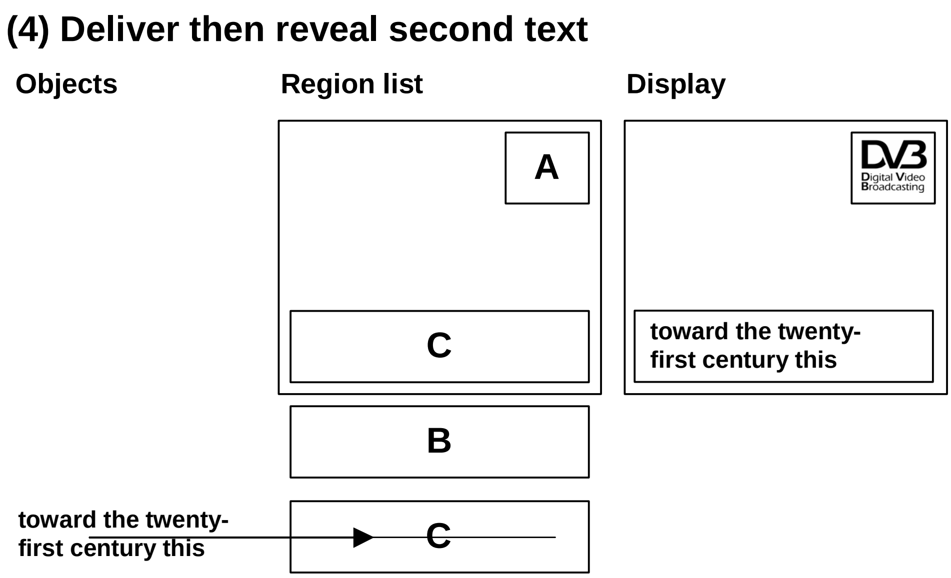


Figure A.4: Deliver then reveal second text

(5) Deliver then reveal third text

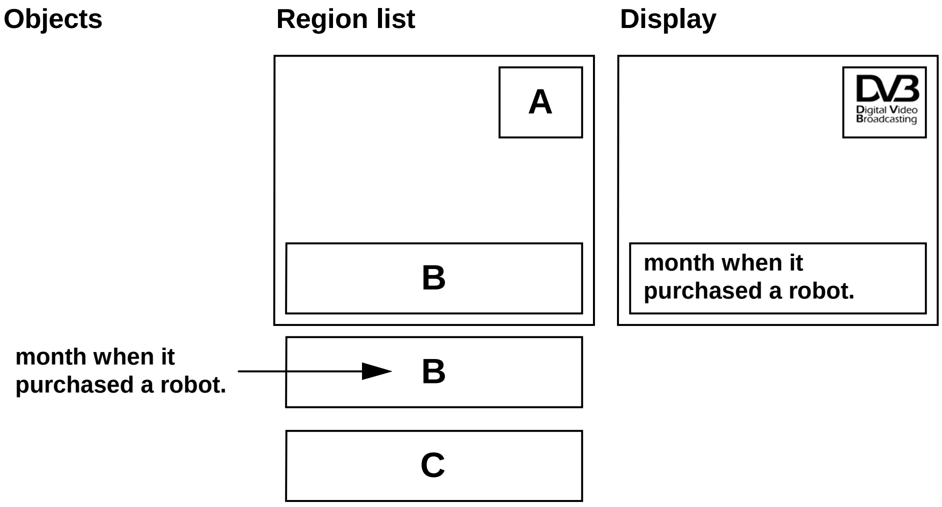


Figure A.5: Deliver then reveal third text

A.5.1.2 Stenographic subtitles

Four regions are defined (A, B, C, D). Regions A, B, C and D are identically sized rectangles sufficient to display a line of text each.

Initially the region list is A, B and C which are presented adjacent to each other to provide a 3-line text console. This region list is used for several page instances as new words are broadcast progressively filling A then B and finally C. When region C has been filled the region list for subsequent page instances uses B, C and D. In effect the text console has been scrolled-up by one line so region A is now available again for new text. This process can continue with every few page instances the region list being changed to scroll the console (e.g. A, B and C then B, C and D then C, D and A).

Figure A.6: Initial display (empty)

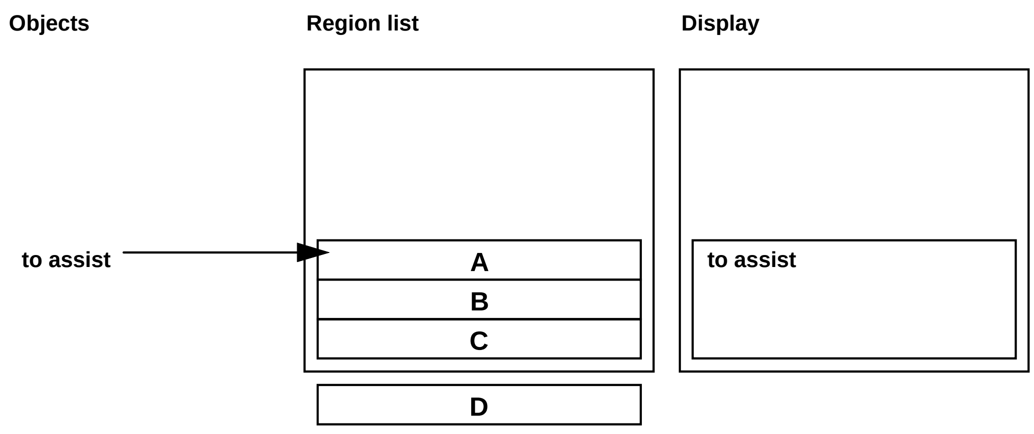


Figure A.7: Introduce regions and deliver and reveal first text

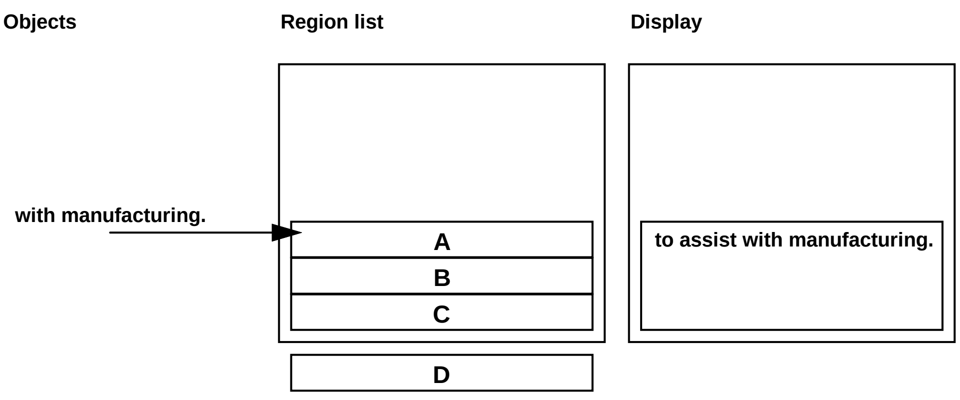


Figure A.8: Deliver update to region A

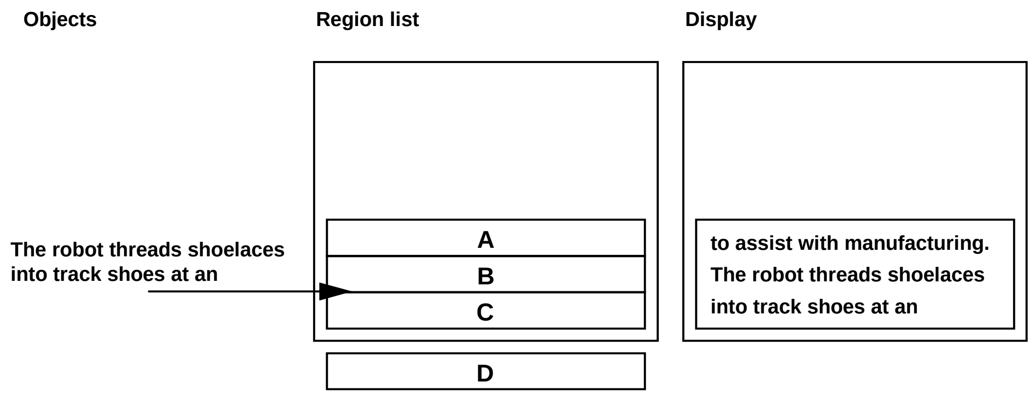


Figure A.9: Deliver updates to regions B and C

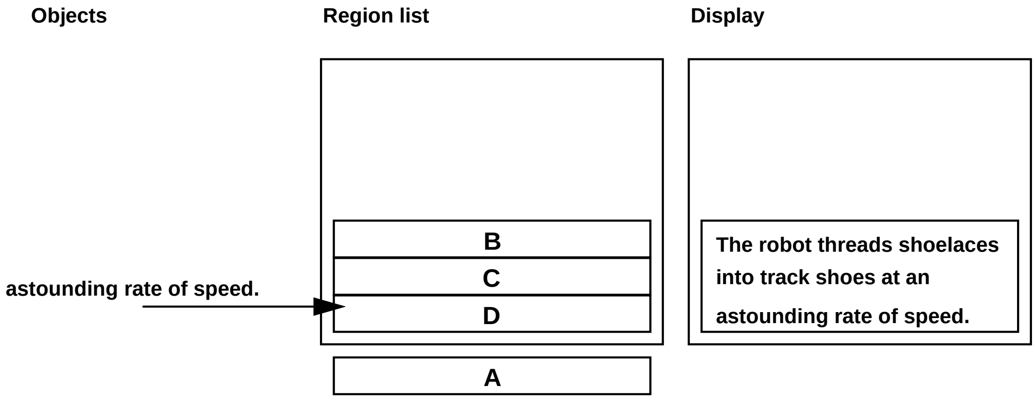


Figure A.10: Scroll regions and deliver update to region D

Annex B (informative): Use of the DDS for SDTV, HDTV and UHDTV services

B.1 Introduction

This annex illustrates approaches to the use of the display definition segment for DTV services through worked examples.

B.2 SDTV services

DVB subtitles for an SDTV service can be coded according to this syntax in one of two ways:

- The display_definition_segment is omitted and the stream encoded on the assumption that the display is 720 pixels by 576 lines (i.e. as per ETSI EN 300 743 (V1.2.1) [5]).
- A display_definition_segment is included in the stream with the signalled values of display_width set to 719 and display_height to 575. The display_window_flag is set to 0 indicating that the display and subtitle window are the same. No display window parameters are transmitted.

B.3 HDTV services

Three worked examples are provided for use of the display_definition_segment with HDTV services:

a) DVB subtitles for a 1 920 by 1 080 pixels HDTV service with no constraints:

  - A display_definition_segment is included in the stream with the signalled values of display_width set to 1 919 and display_height to 1 079. The display_window_flag is set to 0 indicating that the display and subtitle window are the same. No display window parameters are transmitted.

b) DVB subtitles for an HDTV service where the on-screen graphics display is standard definition (720 by 576 pixels) and is upconverted by the IRD before being overlaid on the HDTV video image:

  - The display_definition_segment is omitted and the stream encoded as per ETSI EN 300 743 (V1.2.1) [5].
  - A display_definition_segment is included in the stream with the signalled values of display_width set to 719 and display_height to 575. The display_window_flag is set to 0 indicating that the display and subtitle window are the same. No display window parameters are transmitted.

c) DVB subtitles for a 1 920 by 1 080 pixels HDTV service generated as SDTV-resolution subtitles and constrained to be rendered in the centre 720 pixels horizontally and bottom 576 lines vertically:

  - A display_definition_segment is included in the stream with the signalled values of display_width set to 1 919 and display_height to 1 079. The display_window_flag is set to 1 indicating that the display and subtitle window are not the same. The display window parameters signalled are as follows:
    - display_window_horizontal_position_minimum = 600
    - display_window_horizontal_position_maximum = 1 319
    - display_window_vertical_position_minimum = 504
    - display_window_vertical_position_maximum = 1 079 (see note)

NOTE: Unless the subtitle stream is to be shared by simulcast HDTV and SDTV services, with example c) there is no need to worry about graphics safe areas in the SD stream so the whole 720 by 576 pixels image area can be used for subtitles.

B.4 UHDTV services

Two worked examples are provided for use of the display_definition_segment with UHDTV services:

a) DVB subtitles for UHDTV services are provided in HDTV spatial resolution (1 920 by 1 080 pixels). The UHDTV IRD is expected to upconvert subtitle images before overlaying them on the UHDTV video image:

- A display_definition_segment is included in the stream with the signalled values of display_width set to 1 919 and display_height to 1 079. The display_window_flag is set to 0 indicating that the display and subtitle window are the same. No display window parameters are transmitted.

b) DVB subtitles for a 3 840 by 2 160 pixels UHDTV service generated as HD-resolution subtitles and constrained to be rendered in the centre 1 920 pixels horizontally and bottom 1 080 lines vertically:

- A display_definition_segment is included in the stream with the signalled values of display_width set to 3 839 and display_height to 2 159. The display_window_flag is set to 1 indicating that the display and subtitle window are not the same. The display window parameters signalled are as follows:

    - display_window_horizontal_position_minimum = 960
    - display_window_horizontal_position_maximum = 2 879
    - display_window_vertical_position_minimum = 1 080
    - display_window_vertical_position_maximum = 2 159

Annex C (informative): Illustration of the application of the disparity_shift_update_sequence mechanism for 3D content

The example shown in figure C.1 contains two regions (region1 and region2), each of which has a single subregion equal in size to the region itself.

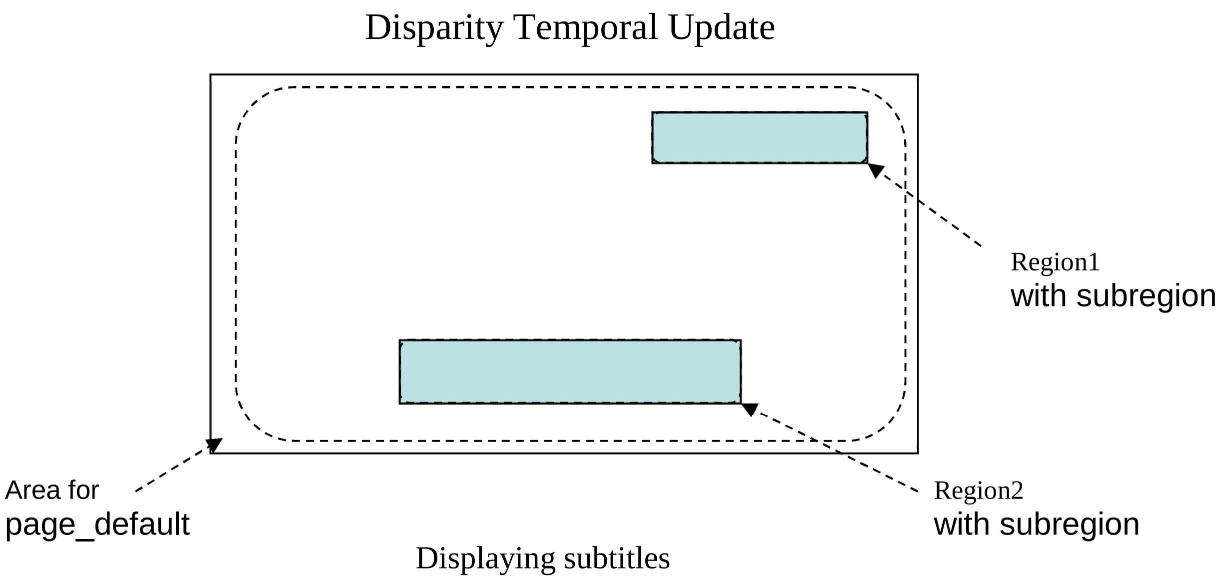
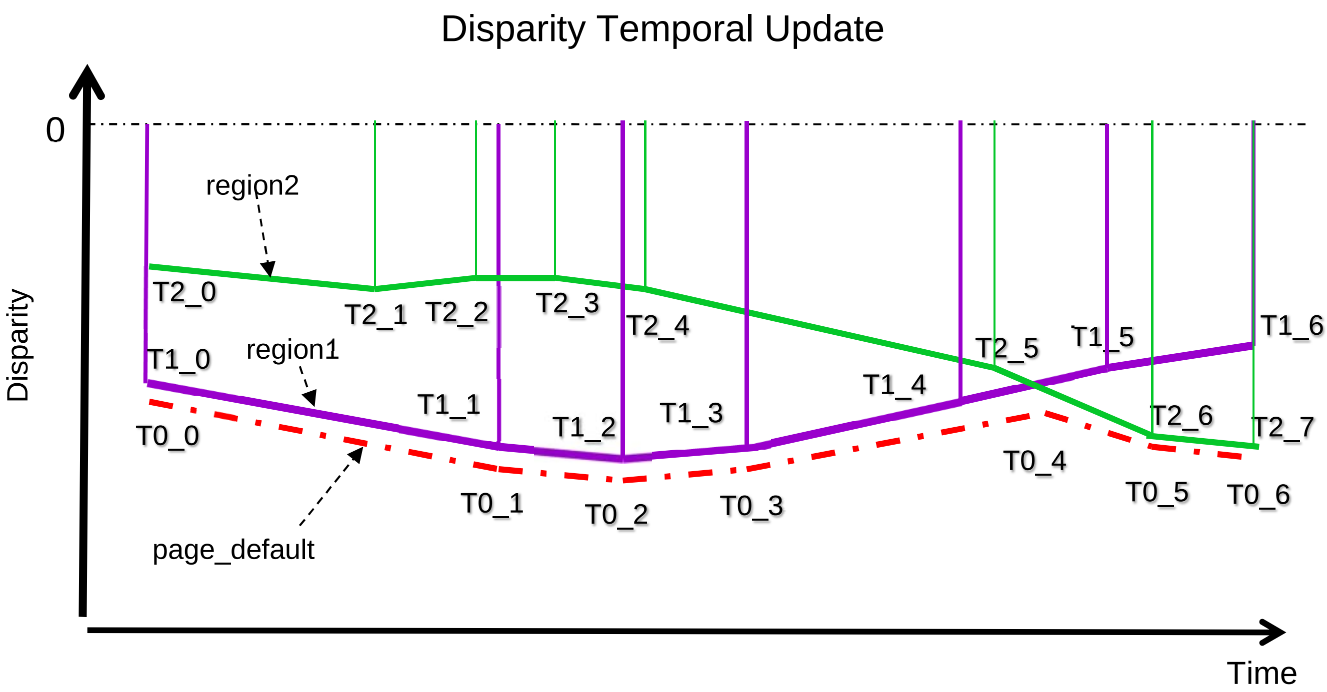


Figure C.1: Example of disparity update applying to the page default and to 2 regions

Figure C.2 depicts the variation of display shift update values in the present example.



NOTE 1: Disparity_shift_update_time Tm_n is expressed as:
Equation E.1: Tm_n = Tm_(n-1) + (interval_duration x interval count) where Tm_0 = PTS in PES header.

NOTE 2: In (T1_n-1, T1_n), the intermediate values between the vertices are generated by decoder interpolation.

NOTE 3: The signalled page default disparity values are calculated by the encoder.

Figure C.2: Disparity shift update values applied to example

From equation E.1 in figure C.2, each disparity update timing Tm_n is calculated by multiplying the interval_duration by the interval_count and adding it to the previous update timing Tm_(n-1). The period between Tm_(n-1) and Tm_n is interpolated by the decoder.

The update timing Tm_n of each region may be independent and is set by the encoder. The example shown in figure C.2 has two regions and a page default disparity update sequence. Region 1's disparity shift update sequence starts from T1_0 with successive updates for T1_1, T1_2 .. T1_6. Region 2's disparity shift update sequence starts from T2_0 with successive updates for T2_1, T2_2 .. T2_7. The page default disparity shift update sequence starts from T0_0 with successive updates for T0_1, T0_2 .. T0_6.

The number of updates differs between the page default, region1 and region2 but the timing of the end of the sequence is the same. The page default disparity shift value would typically be created by taking the minimum value at the corresponding time stamp of all the regions. Figure C.3 shows the hierarchy of the disparity update data structure within the disparity_shift_update_sequence.

Page layer

- page_default_disparity_shift
- T0_0 : interval_count
  disparity_shift_page_update

:

- T0_6 : interval_count
  disparity_shift_page_update

Region layer region1 (subregion1)

- subregion_disparity_shift_integer_part
  subregion_disparity_shift_fractional_part
- T1_0 : interval_count
  disparity_shift_region_update_integer_part
- T1_6 : interval_count
  disparity_shift_region_update_integer_part

Region layer region2 (subregion1)

- subregion_disparity_shift_integer_part
  subregion_disparity_shift_fractional_part
- T2_0 : interval_count
  disparity_shift_region_update_integer_part
- T2_7 : interval_count
  disparity_shift_region_update_integer_part

Figure C.3: Overview of the structure of a disparity_shift_update_sequence

Timing Constraints:

1) Every disparity_shift_update_sequence should be received in the decoder's compressed buffer prior to the presentation time of the corresponding subtitle display set.

2) The time interval between the successive disparity updates should be greater than or equal to 33 ms, which corresponds to a frame rate of 30 Hz or less, or greater than or equal to 40 ms for 25 Hz systems.

3) Disparity update mechanism:

   Division_Period_n = interval_duration × (variable value)

   In the interval (T1_n-1, T1_n), the intermediate values may be generated through interpolation.

NOTE: Disparity_shift_update_time Tm_n is expressed as:

- Tm_n = Tm_(n-1) + (interval_duration × interval_count)
- where Tm_0 = PTS in PES header.

Concurrently, the initial disparity value in the disparity shift update sequence is encoded with the interval_count being set to 0.

Compliant decoder:

4) All decoders should decode the disparity shift update sequence if the disparity_shift_update_sequence_page_flag is set to "1". In this case the decoder should ignore the page_default_disparity_shift and apply to the page the disparity values signalled in the relevant disparity_shift_update_sequence.

5) High performance decoders should decode the disparity shift update sequence if the disparity_shift_update_sequence_region_flag is set to "1". In this case the decoder should ignore the subregion_disparity_shift values and apply to each subregion the disparity values signalled in the relevant disparity_shift_update_sequence.

Other:

6) A disparity update trajectory is created in the decoder from the successive disparity values contained within a display_shift_update_sequence. Interpolation may be applied to generate intermediate disparity values as illustrated by the dotted line in figure C.4. Such interpolation is beneficial but is optional.

7) If the cumulative disparity sequence duration is shorter than the subtitle display set lifetime the decoder should use the last signalled values of disparity until the end of presentation of the display set.

   If the cumulative disparity sequence duration is longer than the subtitle display set lifetime the decoder should ignore those signalled disparity values. which would apply beyond the lifetime of the display set.

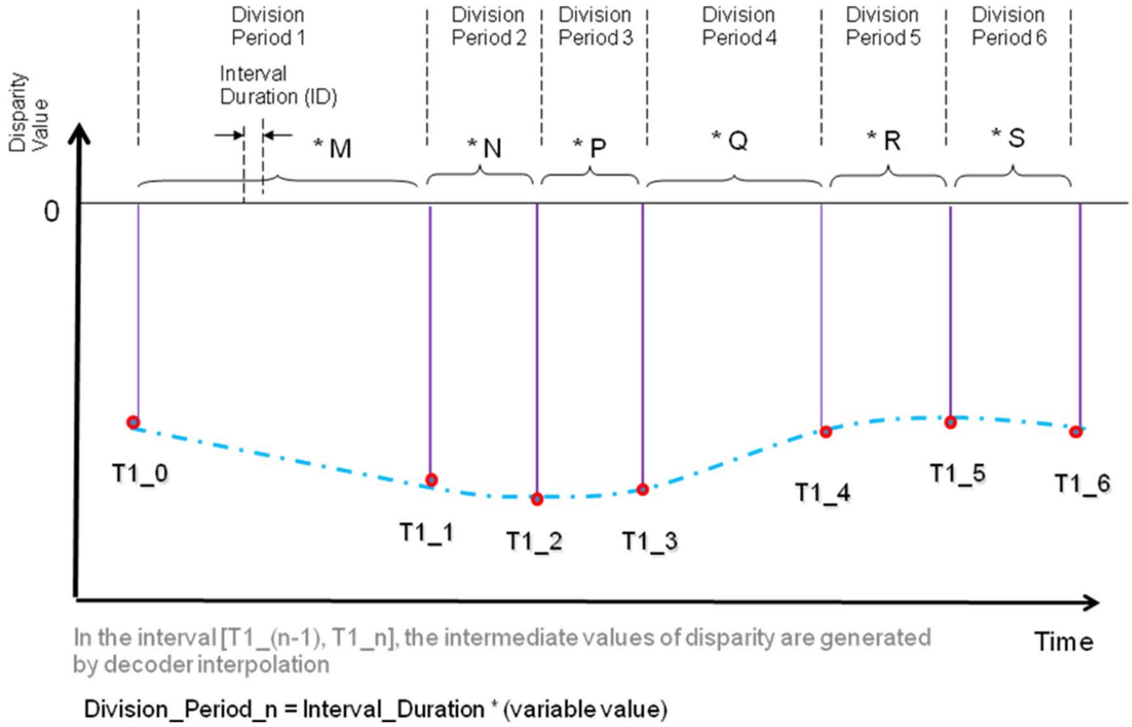


Figure C.4: Disparity update sequence showing interpolation

Annex D (informative): Guidelines on the use of ETSI EN 300 743 for 3D content

General Guidelines

1) To maintain visual integrity, 3D AV services should always be delivered with subtitles coded to include a DSS containing appropriate disparity values. Delivery of 3D content with only 2D subtitles or with a DSS encoding a static zero disparity is firmly discouraged (see also 23) below).

2) It is strongly recommended that a 3D IRD always decodes the DSS and applies the disparity values signalled therein to the subtitles as presented.

3) As noted in clause 7.2.7, it is important to avoid overshoot in the interpolated values of disparity particularly for positive values. One straightforward means of achieving this is to apply linear interpolation between the signalled disparity values.

4) Where an AV service offers several languages of subtitles there are a number of practical reasons why delivering these on separate PIDs is strongly preferred particularly if one language is to be dropped at a network/delivery-platform boundary (see figure 2). From long experience the use of separate PIDs is already the norm.

For 3D services with multiple language subtitles, it is strongly recommended that each language be delivered as a separate subtitle stream and on a separate PID.

5) Because the DSS contains supplementary information relating to the active regions the natural position for a DSS in a subtitling stream is immediately following the RCSs. In order to maintain the logical temporal delivery of the subtitle data set it is strongly recommended that this approach is followed when encoding streams with a DSS.

Alternative strategies are however feasible; decoder designers should therefore ensure that their subtitle decoding software is not sensitive to the particular order of segments within a subtitle display set.

6) A DSS should carry a single disparity value only when used for the static placement of a subtitle over the display period. Subtitles should not be updated dynamically using successive single disparity values (see 11) below).

Subregions

7) Any region declared in a Region Composition Segment to which a disparity is to be ascribed should always have one or more subregions declared in the associated DSS.

8) To achieve a satisfactory user experience all IRDs should use the subregion disparity data declared in the "while" loop of the DSS.

NOTE: Less-sophisticated IRDs may use the page default disparity applied to all active regions or subregions; however doing so will result in a substandard user experience.

9) It is strongly recommended that encoders ascribe suitable values to the page disparity and, as appropriate, its associated update sequence (e.g. copying the values signalled for the forward-most subregion in the "while" loop).

10) Note that the maximum number of subregions assigned to any PTS should be constrained to 4.

Disparity Updates

11) If the subtitle depth changes dynamically during the display period it is strongly recommended that the **disparity_shift_update_sequence** be used in the encoded stream.

12) If the **disparity_shift_update_sequence_region_flag** is set, it is strongly recommended that any fully-capable decoder interprets and makes use of the **disparity_shift_update_sequence** in any display set rather than applying the values encoded in the **subregion_disparity_shift_integer_part** and in the **subregion_disparity_shift_fractional_part**.

13) The **disparity_shift_update_sequence** is applied equally to the page and subregion. If a subregion includes a **disparity_shift_update_sequence**, then the page which contains the subregion should also include a **disparity_shift_update_sequence**, and vice versa.

14) To minimize encoding and decoding delay, the **disparity_shift_update_sequence** can be divided into multiple sets.

15) Values encoded in successive **disparity_shift_update_sequences** should be consistent with the intended dynamic of the displayed disparity. Thus, if a continuous sequence of disparity values is segmented across more than one update sequence the first value signalled in the next sequence should be the same as the final value signalled in the previous one (see figure D.1).

16) It is strongly recommended that within the **disparity_shift_update_sequence**, the **division_period_count** be given a value less than or equal to 16.

Timing

17) It is strongly recommended that each DSS be delivered to the decoder buffer at least 300 ms prior to the validity of its PTS.

18) It is strongly recommended that, when multiple DSSs are employed to dynamically update a subtitle display set, successive DSSs are separated by a minimum period of 200 ms.

19) Within the **disparity_shift_update_sequence** structure, the **interval_count** specifies the multiplier used to calculate the PTS for this disparity update from the initial PTS value.

The calculation for the PTS for this update is:

$$PTS_{new} = PTS_{previous} + (interval\_duration \times interval\_count)$$

where (a) interval_count ≥ 0, (b) PTS_{new} increases with every iteration of the loop & (c) the initial value of $PTS_{previous}$ is the PTS signalled in the PES header.

20) AV material is sometimes originated and coded for delivery at one frame-rate and then reused in another region or country with a different frame-rate. Independent of any standards conversion applied to the video, in some such instances the subtitle stream component is simply re-multiplexed with a modified timestamp. If this is the case, the interpolation of disparity values signalled in the DSS needs to be adjusted to suit the required output video display rate. This can be achieved by inferring the original frame rate from inspection of values of **interval_duration** because the value signalled corresponds to an exact number of frame periods for the original signal.

21) In the first entry of any **disparity_shift_update_sequence**, the value of **interval_count** should be set to 0 if the **disparity_shift_update_integer_part** is intended to be timed to the PTS value.

22) Following a shot change it takes the viewer a short time to "take in" salient detail of the new scene. With 3D content this will include adapting to the disparity of the new dominant features. It is therefore recommended that, when appropriate, subtitles are authored so as to commence a short time (e.g. $^1/_2$ - 1 second) after the shot-change.

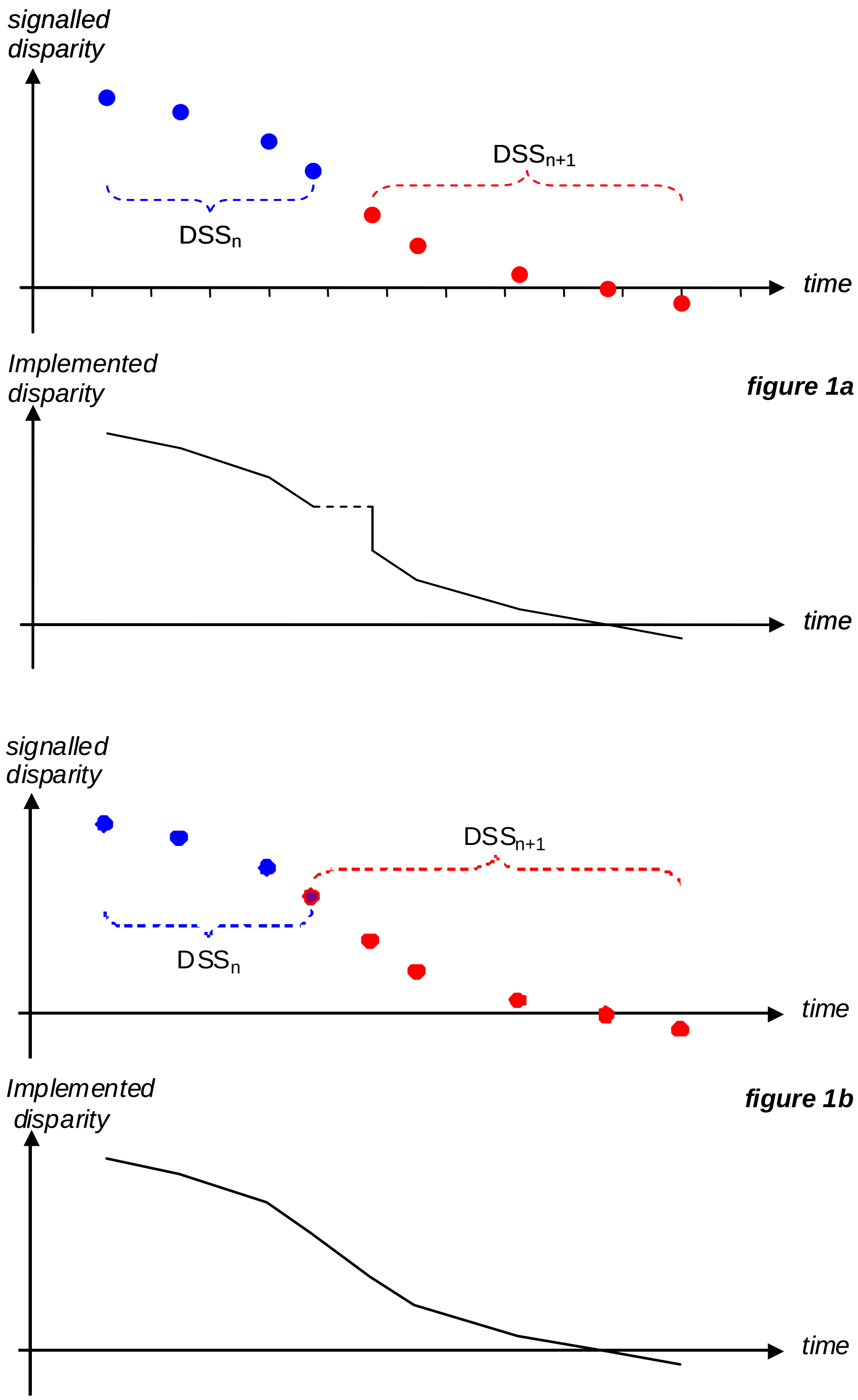
    There are equally occasions when dialogue continues across a shot-change; in such a case care should be taken to assign the appropriate values of disparity to the associated subtitles across the shot change.

    This reinforces the importance of respecting the PTS of the subtitle stream and of accurately maintaining the timing of the encoded and decoded subtitles.

Legacy IRDs

23) Experiments have shown that a number of legacy 2D/HD IRDs do not behave in a predictable and user-friendly manner when presented with subtitle streams containing a DSS.

Broadcasters, service providers and network operators should note that services intended for 2D IRDs but derived from 3D services should therefore include subtitle streams coded in accordance with ETSI EN 300 743 (V1.3.1) [6] i.e. without a DSS. In the case of service-compatible 3D this may involve providing two subtitle streams per language carried on separate PIDs (one with and one without a DSS) and distinguishing the 2D and 3D versions of the service appropriately in the PSI.



1a - last value of DSS_n is held until the time of first value of DSS_{n+1}.
1b - last value of DSS_n and first value of DSS_{n+1} correspond in timing and value.

Figure D.1: Maintaining consistency in successive disparity shift update sequences

Annex E (informative): Conversion of PNG files into progressively coded bitmap subtitles

The PNG format [16] is a graphics format that is in wide use for many applications, including subtitles or captions. As specified in clause 7.2.5.3, the present document adopts the same compression method as used in the PNG format for the carriage of progressively-coded bitmap subtitles in the object data segment, with object coding method = 2 (progressive coding of pixels). Thus, while direct usage of PNG bitmap subtitle files is not foreseen in the DVB subtitling system, PNG files that are generated with suitable parameters can be converted very conveniently into progressively-coded bitmap subtitles. In this way, existing workflows for the creation and delivery of subtitles based on the PNG format can easily be extended to provide subtitles into a deployment using DVB subtitling. Conversely, the specified usage of PNG compression enables efficient carriage of subtitles originating in the PNG format, within the existing DVB subtitling framework (display and region management, segment structure, etc.).

The following is the list of PNG parameters that enable the PNG file to be re-used, after a simple conversion operation, in the DVB subtitling system:

- Bit depth = 8.
- Colour type = 3, for indexed colour.
- Compression method = 0, for DEFLATE [12].
- Filter method = 0, for adaptive filtering with five basic filter types.
- Interlace method = 0, for no interlace.

A DVB CLUT definition segment can be generated from the PNG file by extracting the CLUT entries from the *PLTE* (palette) chunk and converting them to the equivalent Y, Cr and Cb values for the corresponding CLUT entry in the DVB subtitling system. Alpha values for the CLUT definition segment can be taken from the *tRNS* (transparency) chunk, if provided, and used as the T-values for the corresponding CLUT entries in the DVB CLUT.

The compressed image data for a progressively-coded DVB subtitle is obtained by extracting the PNG image data from the one or more IDAT chunks and storing the raw *zlib* data as the *compressed_bitmap_data_byte*s in the *progressive_pixel_block* structure. The *bitmap_width* and *bitmap_height* fields can be extracted from the PNG IHDR chunk fields *width* and *height* respectively. These values should never need more than 16 bits since the source files will be of a suitable size for DVB subtitles. The DVB subtitle encoder then calculates the *compressed_data_block_length* based on the size of the raw compressed bitmap byte array.

Figure E.1 provides an overview of such a conversion process.

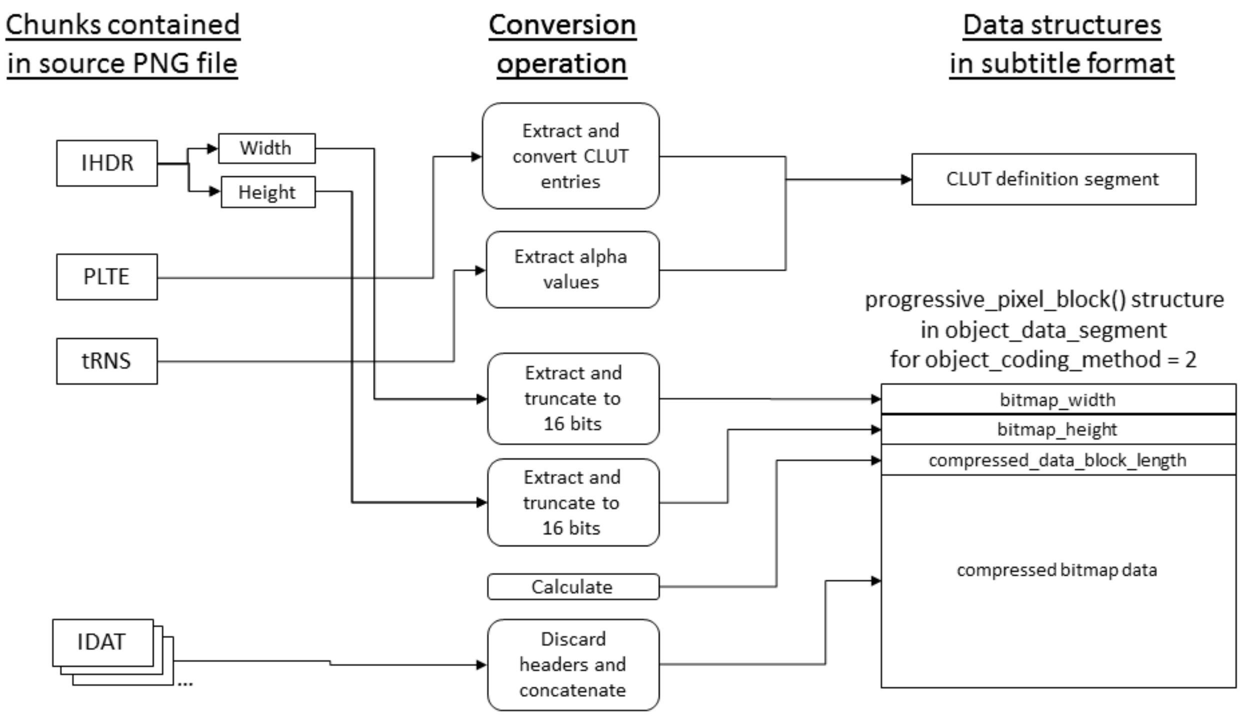


Figure E.1: PNG file conversion process

Annex F (informative): Bibliography

ISO/IEC 10646-1: "Information technology - Universal Multiple-Octet Coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane".

History

Document history				
Edition 1	September 1997	Publication as ETSI ETS 300 743		
V1.2.1	October 2002	Publication		
V1.3.1	November 2006	Publication		
V1.4.1	October 2011	Publication		
V1.5.1	January 2014	Publication		
V1.6.1	July 2018	EN Approval Procedure	AP 20181022:	2018-07-24 to 2018-10-22
V1.6.1	October 2018	Publication		