ENGLISH TRANSLATION

MMT-BASED MEDIA TRANSPORT SCHEME IN DIGITAL BROADCASTING SYSTEMS

ARIB STANDARD

ARIB STD-B60 Version 1.13

Version 1.13 October 11, 2018

Association of Radio Industries and Businesses

General Notes to the English Translation of ARIB Standards and Technical Reports

1. Notes on Copyright

- The copyright of this document is ascribed to the Association of Radio Industries and Businesses (ARIB).
- All rights reserved. No part of this document may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, without the prior written permission of ARIB.

2. Notes on English Translation

- ARIB Standards and Technical Reports are usually written in Japanese. This document is a translation into English of the original document for the purpose of convenience of users. If there are any discrepancies in the content, expressions, etc. between the original document and this translated document, the original document shall prevail.
- ARIB Standards and Technical Reports, in the original language, are made publicly available through web posting. The original document of this translation may have been further revised and therefore users are encouraged to check the latest version at an appropriate page under the following URL:
  http://www.arib.or.jp/english/index.html.

Foreword

The Association of Radio Industries and Businesses (ARIB) investigates and summarizes the basic technical requirements for various radio systems in the form of "ARIB Standards". These standards are developed with the participation of and through discussions amongst radio equipment manufacturers, telecommunication operators, broadcasting equipment manufacturers, broadcasters and users.

ARIB Standards include "government technical regulations" (mandatory standard) that are set for the purpose of encouraging effective use of frequency and preventing interference with other spectrum users, and "private technical standards" (voluntary standards) that are defined in order to ensure compatibility and adequate quality of radio equipment and broadcasting equipment as well as to offer greater convenience to radio equipment manufacturers, telecommunication operators, broadcasting equipment manufacturers, broadcasters and users.

This ARIB Standard is developed for transmission system for digital terrestrial television broadcasting. In order to ensure fairness and transparency in the defining stage, the standard was set by consensus at the ARIB Standard Assembly with the participation of both domestic and foreign interested parties from radio equipment manufacturers, telecommunication operators, broadcasting equipment manufacturers, broadcasters and users.

ARIB sincerely hopes that this ARIB Standard will be widely used by radio equipment manufacturers, telecommunication operators, broadcasting equipment manufacturers, broadcasters and users.

NOTE:
Although this ARIB Standard contains no specific reference to any Essential Industrial Property Rights relating thereto, the holders of such Essential Industrial Property Rights state to the effect that the rights listed in the Attachment 1 and 2, which are the Industrial Property Rights relating to this standard, are held by the parties also listed therein, and that to the users of this standard, in the case of Attachment 1, such holders shall not assert any rights and shall unconditionally grant a license to practice such Industrial Property Rights contained therein, and in the case of Attachment 2, the holders shall grant, under reasonable terms and conditions, a non-exclusive and non-discriminatory license to practice the Industrial Property Rights contained therein. However, this does not apply to anyone who uses this ARIB Standard and also owns and lays claim to any other Essential Industrial Property Rights of which is covered in whole or part in the contents of the provisions of this ARIB Standard.

ARIB STD-B60
Version 1.13-E1

Attachment 1 (Selection of Option 1)
(N/A)

Attachment 2 (Selection of Option 2)

Patent Applicant/ Holder	Name of Patent	Registration No./ Application No.	Remarks
Mitsubishi Electric Corporation	Coding Equipment Note 1	Application No. 2014-112114	JP
Japan Broadcasting Corporation (NHK)	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.0 Note1		
Sony Corporation	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.0 Note 1		
Sharp Corporation	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.0 Note 1		
Panasonic Corporation	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.0 Note 1		
	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.6 Note 3		
QUALCOM Incorporated	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.0 Note 1		
Japan Broadcasting Corporation (NHK)	Submitted comprehensive confirmation of patents for ARIB STD-B60 Ver1.2 Note 2		

Note 1 : Valid for ARIB STD-B60 Ver1.0 (received on July 24, 2014)
Note 2 : Valid for revised part of ARIB STD-B60 Ver1.2 (received on March 10, 2015)
Note 3 : Valid for revised part of ARIB STD-B60 Ver1.6 (received on March 18, 2016)

CONTENTS

Chapter 1: General Terms

1.1 Purpose

The purpose of this standard is to prescribe the transport system of video, audio, data, etc. by MMT for digital broadcasting.

1.2 Scope

This standard applies to digital broadcasting which uses MMT.

1.3 References

1.3.1 Normative References

The following documents are those in which some of the items provided in the documents are quoted in this standard.

(1) Ordinance of the Ministry of Internal Affairs and Communications No.87, 2011: "Standard transmission system for digital broadcasting among standard television broadcasting and the like" (Partial Amendment: Dec. 10, 2013, July 3, 2014, and July 29, 2016. Hereinafter referred to as "Ordinance.")

(2) Notification of the Ministry of Internal Affairs and Communications No.233, 2014: "On defining the configuration of related information and the procedure of transmission, the procedure of transmission for PES packet, section format, TS packet, IP packet, ULE packet, MMTP packet, compressed IP packet and TLV packet, the configuration of pilot signal and identifier, and the configuration of emergency information descriptor and message of emergency alarm broadcasting" (hereinafter referred to as "Notification.")

(3) ISO/IEC 23008-1:2017: Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 1: MPEG media transport (MMT) (hereinafter referred to as "MMT Standard.")

(4) Rec. ITU-T H.222.0|ISO/IEC 13818-1:2017: Information technology — Generic coding of moving pictures and associated audio information: Systems

(5) ISO/IEC 14496-12:2015: Information technology — Coding of audio-visual objects — Part 12: ISO base media file format

(6) ISO/IEC 10646:2012: Information technology — Universal Coded Character Set (UCS)

(7) IETF RFC 768: User Datagram Protocol, Aug. 1980

(8) IETF RFC 791: Internet Protocol, Sep. 1981

(9) IETF RFC 2460: Internet Protocol, Version 6 (IPv6) Specification, Dec. 1998

(10) IETF RFC 5905: Network Time Protocol Version 4: Protocol and Algorithms Specification, June 2010

(11) Rec. ITU-R BT.1869-0 (2010): Multiplexing scheme for variable-length packets in digital multimedia broadcasting systems

1.3.2 Informative References

Standards that are related to this standard are as follows.

(1) ISO/IEC 23008-2:2017: Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding

(2) ISO/IEC 14496-3:2009: Information technology — Coding of audio-visual objects — Part 3: Audio

(3) ARIB STD-B10: “Service Information for Digital Broadcasting System”

(4) ARIB STD-B32: “Video Coding, Audio Coding and Multiplexing Specifications for Digital Broadcasting”

(5) ARIB STD-B61: “Conditional Access System (Second Generation) and CAS Program Download System Specifications for Digital Broadcasting”

(6) ARIB STD-B62: “Multimedia Coding Specification for Digital Broadcasting (Second Generation)”

1.4 Abbreviations

AAC	Advanced Audio Coding
AIT	Application Information Table
AL-FEC	Application Layer Forward Error Correction
ALS	Audio Lossless Coding
AMT	Address Map Table
BIT	Broadcaster Information Table
CA	Conditional Access
CDT	Common Data Table
CRC	Cyclic Redundancy Check
CRI	Clock Relation Information
DCI	Device Capability Information
DCM	Download Control Message
DMM	Download Management Message
ECM	Entitlement Control Message
EIT	Event Information Table
EMM	Entitlement Management Message
EPG	Electronic Program Guide
EXI	Efficient XML Interchange
GFD	Generic File Delivery
HCfB	Header Compression for Broadcasting

HEVC	High Efficiency Video Coding
HLG	Hybrid Log-Gamma
HRBM	Hypothetical Receiver Buffer Model
HTML	Hyper Text Markup Language
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISO	International Organization for Standardization
LCT	Layout Configuration Table
LDT	Linked Description Table
MAC	Media Access Control
MFU	Media Fragment Unit
MMT	MPEG Media Transport
MMTP	MMT Protocol
MPI	MMT Presentation Information
MPT	MMT Package Table
MPU	Media Processing Unit
MTU	Maximum Transmission Unit
NIT	Network Information Table
NPT	Normal Play Time
NTP	Network Time Protocol
PA	Package Access
PLT	Package List Table
PQ	Perceptual Quantization
PU	Presentation Unit
RFC	Request For Comment (IETF standard)
SDT	Service Description Table
SDTT	Software Download Trigger Table
SI	Signaling Information
TCP	Transmission Control Protocol
TLV	Type Length Value
TMCC	Transmission and Multiplexing Configuration Control
TTML	Timed Text Markup Language
UDP	User Datagram Protocol
URL	Uniform Resource Locator

Chapter 2: Broadcasting System Using MMT

2.1 Protocol Stack of Broadcasting System Using MMT

Protocol stack of broadcasting system using MMT is shown in Fig. 2-1.

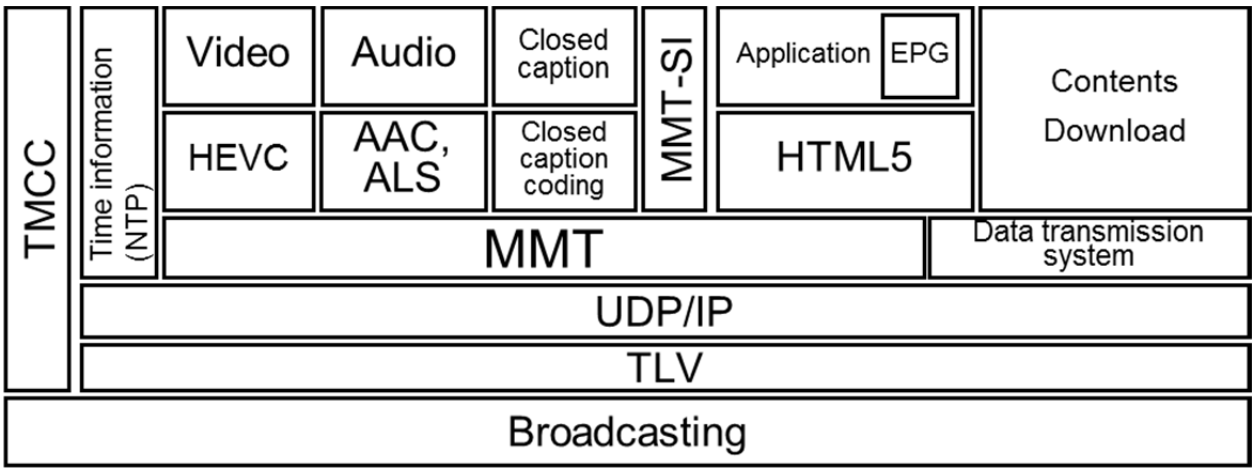


Fig. 2-1: Protocol stack of broadcasting system using MMT

Code of video signal and audio signal for broadcasting program shall be MFU/MPU, and it is packetized in MMTP (MMT Protocol) payload, and transmitted by IP packet. Also, data contents and closed caption signals which are related to broadcasting program shall be MFU/MPU format, and packetized in MMTP payload, and transmitted by IP packet. There are four kinds of transmission system shown in Table 2-1 for data content transmission. Data is packetized in MMTP and transmitted by IP packet in any transmission system.

Table 2-1: Kinds of data transmission system

Data transmission system	Summary of function and main usage
Closed caption / superimposition Transmission system	This is used for streaming of data which is synchronized with broadcast program. It is suitable for transmission of closed caption and super imposition.
Application Transmission system	This is used for streaming of data which is not synchronized with broadcast program. It is suitable for transmission of data which is used for download, multimedia service, etc.
Event message Transmission system	This is used for a message noticing whether the application of the receiver is synchronous or asynchronous which is sent from broadcasting station. It is used for multimedia service.
General data Transmission system	This is a system which transmits various kinds of data by synchronous type or asynchronous type. It is suitable for streaming of the data which is used in the player which indicates data except video, audio and closed caption, and the data which is used for multimedia service.

Such a composed IP packet is transmitted by TLV packet format in the broadcasting channel. One IP packet or one header-compressed IP packet is transmitted in one TLV packet (ARIB STD-B32).

In addition to transmission system for these media data, two kinds of control information: MMT-SI and TLV-SI are prepared for broadcasting system. MMT-SI is the control information which represents a construction of broadcasting program, etc. It is formed by the control message of MMT, and being conveyed by MMTP payload and MMTP packetized, it is transmitted by IP packet. TLV-SI is the control information on multiplex of IP packet, and provides information for tuning and corresponding information between IP address and service. The process of tuning channel in broadcasting system using MMT is recorded in ARIB STD-B32 Part 1, Attachment: "Restrictions on HEVC parameters and operational guidelines". Moreover, in broadcasting system, time information is also transmitted in broadcasting channel in order to provide absolute time.

In communication channel, media data is transmitted by IP packet according to distribution form of unicast/multicast. Protocol stack in communication channel is shown in Fig. 2-2.

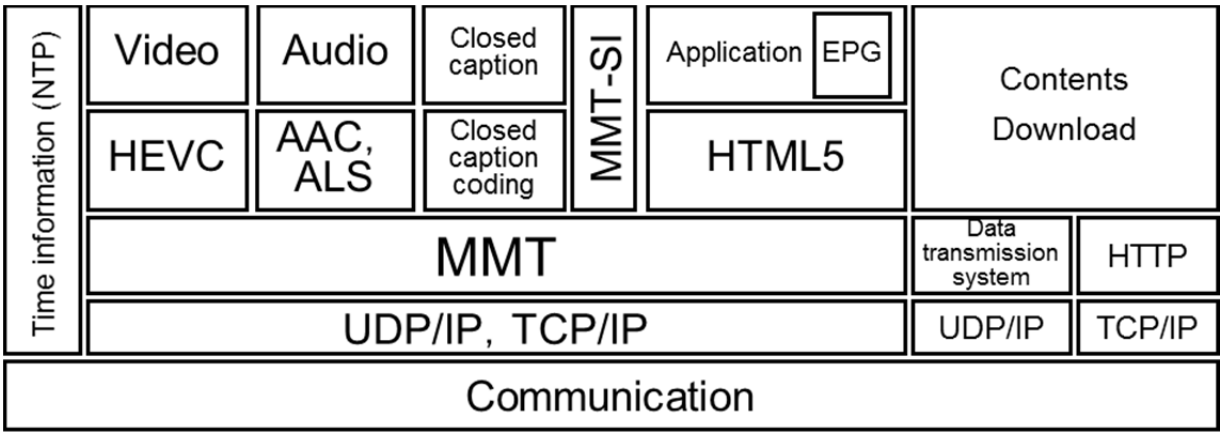


Fig. 2-2 Protocol stack in communication circuit

2.2 Clock Synchronization in Broadcasting System

Broadcasting system using MPEG-2 Systems which is provided in Rec. ITU-T H.220|ISO/IEC 13818-1 has been realized synchronization of video, audio, etc. and stable reproduction on account of the standard. But as MMT standard does not specify the transmission of clock, etc., it is necessary that transmission system and receivers in the broadcasting system using MMT meets the requirements in the following.

- The transmission system must keep clock based on NTP time stamp.
- In the transmission system, the above-mentioned clock must synchronize with the clock which is used for processing video and audio encoding.
- The receiver must regenerate clock based on the received clock information in NTP format, and keep it.
- In the receiver, system clock for keeping above-mentioned clock must synchronize with the clock for processing video and audio encoding.

Chapter 3: Transmission of Time Information by IP Packet

3.1 Configuration of NTP Format

In order to provide Coordinated Universal Time (UTC), IP packet by NTP format which is provided in IETF REC 5905 "Network Time Protocol Version 4: Protocol and Algorithms Specification" is used. Configuration of NTP format is shown in Table 3-1.

Table 3-1 Configuration of NTP format

Data structure	Number of bits	Data notation
Network_Time_Protocol_Data () {		
leap_indicator	2	uimsbf
version	3	uimsbf
mode	3	uimsbf
stratum	8	uimsbf
poll	8	simsbf
precision	8	simsbf
root_delay	32	uimsbf
root_dispersion	32	uimsbf
reference_identification	32	uimsbf
reference_timestamp	64	uimsbf
origin_timestamp	64	uimsbf
receive_timestamp	64	uimsbf
transmit_timestamp	64	uimsbf
}		

Meaning of NTP format:

leap_indicator (leap indicator): This shows to insert or delete leap second into last one minute in the present month, and it is coded according to Table 3-2.

Table 3-2 Leap indicator

Leap indicator	meaning
0	Without alarm
1	Last one minute is 61 seconds.
2	Last one minute is 59 seconds.
3	Alarm

version (version number): This indicates version of NTP. For NTP version 4, this shall be 4.

mode (working mode): This represents working mode of NTP, and coded according to Table 3-3. For broadcasting system, broadcast mode is used as the working mode.

Table 3-3 Working mode

working mode	meaning
0	reserved
1	objective active mode
2	objective passive mode
3	client
4	server
5	broadcast
6	message for NTP control
7	reserved for private use

stratum (stratum): This represents stratum, and is coded according to Table 3-4.

Table 3-4 Stratum

stratum	meaning
0	indefinite or invalid
1	first reference
2 – 15	second references
16	without synchronization
17 – 255	reserved

poll (polling interval): This represents log2 of maximum interval (a unit of second) of successive NTP message.

precision (precision): This represents log2 of precision of system clock (a unit of second).

root_delay (root delay): A shuttle delay till the reference time is represented in NTP short format. The NTP short format is composed of 16 bits field representing a unit of second and 16 bits field representing below one second.

root_dispersion (root dispersion): The dispersion of summed delay to the reference time is represented in NTP short format.

reference_identification (reference identifier): This represents identifier to represent the reference time. For broadcasting system, 0000 representing NULL is stored.

reference_timestamp (reference timestamp): The time when system time was compensated last is represented in NTP long format. NTP long format is composed of 32 bits field representing a unit of second and 32 bits field representing below one second. If MSB of 32 bits representing a unit of second is 0, the year 2036 shall be the reference.

origin_timestamp (origin timestamp): The time of the client when the client transmitted a request to the server is represented in NTP long format. For broadcasting system, 0 is stored.

receive_timestamp (receive timestamp): The time of the server which received a request from the client is represented in NTP long format. For broadcasting system, 0 is stored.

transmit_timestamp (transmit timestamp): The time of the server which transmitted a response to the client is represented in NTP long format. If MSB of 32 bits representing a unit of

second is 0, the year 2036 shall be the reference.

3.2 NTP-Format Transmitting IP Packet

NTP format is transmitted by storing in IP/UDP packet.

For IPv4 packet, the destination address of IPv4 addressee shall be 224.0.1.1. For IPv6 packet, the destination address of IPv6 addressee shall be an IP address which ends at :101. As the number of the addressee port for UDP header, 123 is used.

Also, when IP packet which involves NTP format is transmitted by TLV packet, IP header shall not be compressed.

Chapter 4: Control Information of Broadcasting System Using MMT

The control information has TLV-SI related to the TLV multiplex system for multiplexing IP packet and MMT-SI related to MMT which is the media transport system.

The control information becomes valid from the point in time when the information is received, except the case that an object of the control is expressed clearly (for example, the case that MPU for the object is specified by the MPU sequence number field in the MPU time stamp descriptors, the case of representing that the information will be valid next by the current next indication field, and so on).

4.1 Kinds of TLV-SI

There are tables and descriptors in TLV-SI. The kinds of the tables and the assignment of the table identifiers are shown in Table 4-1 and Table 4-2, respectively, and the kinds of the descriptors and the assignment of the descriptor identifiers are shown in Table 4-3 and Table 4-4, respectively.

Table 4-1 Table name and function of TLV-SI

Table name	Summary of function
Network Information Table for TLV*	For transmission by TLV packet, the information which connects channel information such as modulation frequency with broadcasting program is transmitted.
Address Map Table*	The information which connects service identifier identifying broadcasting program number with IP packet is transmitted.
Table which is prepared by broadcasters	registered and open

*: table which is provided in Ordinance

Table 4-2 Table identifier assignment of TLV-SI

Table identification	Table identification extension	Table name
0x40	—	TLV-NIT (self network)*
0x41	—	TLV-NIT (other network)*
0xFE	0x0000	AMT*

*: according to Notification

Table 4-3 Descriptor name and function of TLV-SI

Descriptor name	Summary of function
Service List Descriptor*1	Description of the program channel and the list of kinds
Satellite Delivery System Descriptor*1	Description of the physical condition of satellite channel
System Management Descriptor*1	Identification of broadcasting or non-broadcasting, etc.
Network Name Descriptor	Description of the network name
Remote Control Key Descriptor	Service is set by which one-touch channel selection buttons are assigned to receiver remote controller.
Table which is prepared by broadcasters	registered and open

*1: descriptor which is provided in Notification

Table 4-4 Assignment of descriptor tag value used for TLV-SI

Descriptor tag value	Descriptor
0x40	Network Name Descriptor
0x41	Service List Descriptor *1
0x43	Satellite Delivery System Descriptor *1
0xCD	Remote Control Key Descriptor
0xFE	System Management Descriptor *1

*1: according to Notification

4.2 Kinds of MMT-SI

There are message, table and descriptor in MMT-SI.

The kinds of the message and the assignment of message identifier are shown in Table 4-5 and Table 4-6, respectively. The kinds of the table and the assignment of table identifier are shown in Table 4-7 and Table 4-8, respectively, and the kinds of the descriptor and the assignment of descriptor identifier are shown in Table 4-9 and Table 4-10, respectively.

Table 4-5 Message name and function of MMT-SI

Message name	Summary of function
Package Access (PA) message*	This transmits the table of MMT-SI, as the entry point of MMT-SI,.
M2 section message*	This transmits the section extension format of MPEG-2 Systems.
CA message*	This transmits the information about conditional access system.
M2 short section message	This transmits the section short format of MPEG-2 Systems.
Data transmission message	This transmits the table about data transmission.
message which is prepared by broadcasters	registered and open

*: descriptor which is provided in Notification

Table 4-6 Assignment of message identifier of MMT-SI

Message ID	Message
0x0000	PA message*1
0x0001 – 0x000F	MPI message*2
0x0010 – 0x001F	MPT message*2
0x0200	CRI message*2
0x0201	DCI message*2
0x0202	AL-FEC message*2
0x0203	HRBM message*2
0x0204 – 0x6FFF	reserved for ISO/IEC (16-bit length message)
0x7000 – 0x7FFF	reserved for ISO/IEC (32-bit length message)
0x8000	M2 section message*1
0x8001	CA message*1
0x8002	M2 short section message
0x8003	Data transmission message
0x8004 – 0xDFFF	reserved (message whose length field is 16 bits) (provided by the Ministry or private standardization organization)
0xE000 – 0xEFFF	message which is prepared by broadcasters (message whose length field is 16 bits)
0xF000 – 0xF7FF	reserved (message whose length field is 32 bits) (provided by the Ministry or private standardization organization)
0xF800 – 0xFFFF	message which is prepared by broadcasters (message whose length field is 32 bits)

*1 : according to Notification
*2 : message which is provided in MMT Standard, but is not used in this standard

Table 4-7 Table name and function of MMT-SI

Table name	Summary of function
MMT Package Table*1	This gives the information composing package, such as a list of asset and its position.
Package List Table	This represents IP data flow and Packet ID which transmit PA message of MMT package that is given as a broadcasting service, and a list of IP data flow which transmits IP service.
Layout Configuration Table	This is used for connecting the layout information for presentation with the layout number.
ECM*1 (Entitlement Control Message)	This transmits program information (information about program and key for descramble, etc.) and common information which is composed of control information.
EMM*1 (Entitlement Management Message)	This transmits the individual information which involves contract information of each subscriber, work key for deciphering cipher with common information, and so on.
CA Table*1 (MH) (Conditional Access Table)	This transmits descriptor about conditional access system.
DCM*2 (Download Control Message)	This transmits the key related information which is composed of key for deciphering channel cipher for download.
DMM*2 (Download Management Message)	This transmits the key related information which is composed of download key for deciphering DCM, etc.
MH-Event Information Table	This transmits information about program such as the name of program, the date and time of broadcasting, explanation of the content, etc.
MH-Application Information Table	This transmits the dynamic control information about application and the information-added which is necessary for execution.
MH-Broadcaster Information Table	This is used for representing the information of broadcaster which exists on the network.
MH-Software Download Trigger Table	This transmits the notice information such as service ID for download, schedule information, the kind of receivers for updating, etc.
MH-Service Description Table	This transmits the information about planning channel such as the name of planning channel, the name of broadcaster, etc.
MH-Time Offset Table	This designates present date and time, and transmits the differential time between actual time and human system.
MH-Common Data Table	This transmits the data which is common to be necessary for receivers such as a logo mark of broadcaster, on a precondition that the data is stored in non-volatile memory of the receiver.
Data Directory Management Table	This gives the directory configuration of file which composes application.
Data Asset Management Table	This gives the MPU configuration in asset and the information of version for each MPU.
Data Content Configuration Table	This gives the information of file configuration as data content.
Event Message Table	This is used for transmitting information about event message.
Table which is prepared by broadcasters	registered and open

*1 : table which is provided in Notification
*2 : table which is provided in ARIB STD-B61

Table 4-8 Assignment of identifier of table of MMT-SI

Table ID	Table name
0x00	PA Table*2
0x01	Subset 0 MPI Table*2
0x02 – 0x0F	Subset 1 MPI Table to subset 14 MPI Table*2
0x10	Complete MPI Table*2
0x11 – 0x1F	Subset 0 MP Table to subset 14 MP Table*1
0x20	Complete MP Table*1
0x21	CRI Table*2
0x22	DCI Table*2
0x23 – 0x7F	reserved for ISO/IEC (16-bit length table)
0x80	PLT
0x81	LCT
0x82 – 0x83	ECM*1
0x84 – 0x85	EMM*1
0x86	CAT (MH)*1
0x87 – 0x88	DCM
0x89 – 0x8A	DMM
0x8B	MH-EIT (present and next program of self-stream)
0x8C – 0x9B	MH-EIT (schedule of self-stream)
0x9C	MH-AIT (AIT controlled application)
0x9D	MH-BIT
0x9E	MH-SDTT
0x9F	MH-SDT (self-stream)
0xA0	MH-SDT (other stream)
0xA1	MH-TOT
0xA2	MH-CDT
0xA3	DDM Table
0xA4	DAM Table
0xA5	DCC Table
0xA6	EMT
0xA7 – 0xDF	Reserved (provided by the Ministry or private standardization organization)
0xE0 – 0xFF	Table which is prepared by broadcasters

*1 : according to Notification
*2 : Table which is provided in MMT Standard, but is not used in this standard

Table 4-9 Descriptor name and function of MMT-SI

Descriptor name	Summary of function
Asset group Descriptor	This provides the group relationship of asset and the priority in the group.
Event package Descriptor	This provides the correspondence between event representing program and package.
Background color designation Descriptor	This designates background color of the deepest background in the layout designation.
MPU presentation region designation Descriptor	This provides MPU presentation position.
MPU time stamp Descriptor*1	This provides MPU presentation time.
Dependency relationship Descriptor*1	This provides asset ID of the asset which is in a dependent relationship.
Access control Descriptor*1	This identifies conditional access system.
Scramble system descriptor*1	This identifies scramble subsystem.
Message certification system Descriptor	This identifies message certification system.
Emergency information Descriptor*1 (MH)	This provides necessary information and description of function as an emergency alarm signal.
MH-MPEG-4 Audio Descriptor	This describes basic information to specify coding parameters of MPEG-4 audio stream.
MH-MPEG-4 Audio extension Descriptor	This describes profile and level of MPEG-4 audio stream and particular set up to the coding system.
MH-HEVC Video Descriptor	This describes basic coding parameters for video stream by ITU-T Recommendation H.265|ISO/IEC 23008-2 (HEVC stream).
MH-Link Descriptor	This describes connection with other organized program channel.
MH-Event group Descriptor	This describes information about grouping of plural events.
MH-Service list Descriptor	This describes a list of organized program channel and the classification.
MH-Short format event Descriptor	This describes the name of program and its simple explanation.
MH-extension format event Descriptor	This describes a detailed information about program.
Video component Descriptor	This describes parameters, explanation, etc. regarding video signal, among the component signals of program.
MH-Stream identification Descriptor	This is used for identifying component signals of each program.
MH-Content Descriptor	This describes the genre of program.
MH-Parental rate Descriptor	This describes an age limit for a viewer.
MH-Audio component Descriptor	This describes the parameters regarding audio signal among program component.
MH-Objective area Descriptor	This describes objective area.
MH-Series Descriptor	This describes the series information over plural events.

Descriptor name	Summary of function
MH-SI transmission parameter Descriptor	This describes the parameters for SI transmission (group of interval, interval of re-transmission, etc.)
MH-Broadcaster name Descriptor	This describes broadcaster name.
MH-Service Descriptor	This describes the name of program channel and the name of broadcasting company.
IP Data flow Descriptor	This describes the information of IP data flow which is involved in service.
MH-CA starting Descriptor	This describes information about starting up CAS program which has conditional access function.
MH-Type Descriptor	This represents the type of file which is transmitted by application transmission system.
MH-Info Descriptor	This describes the information about MPU or item.
MH-Expire Descriptor	This describes the term of validity for the item.
MH-CompressionType Descriptor	This represents the compression algorithm for item which is compressed and transmitted, and the number of Bytes of item before compression.
MH-Data coding system Descriptor	This is used for identifying data coding system.
UTC-NPT reference Descriptor	This transmits the relationship between NPT and UTC.
Event message Descriptor	This transmits the information about event message on the whole.
MH-Local time offset Descriptor	This describes the differential time between the actual time (UTC+9 hours) and the display time for human system when summer time system is excuted.
MH-Component group Descriptor	This describes the information about grouping plural components.
MH-Logo transmission Descriptor	This describes the text sequence for simple logo, pointing for logo by CDT format, and so on.
MPU extension time stamp Descriptor	This provides the decoding time for access unit in MPU and so on.
MPU Download content Descriptor	This describes the attribute information of content which is downloaded by using MPU.
MH-Network download content Descriptor	This describes the attribute information of content which is downloaded by using network.
MH-Application Descriptor	This describes application information.
MH-Transmission protocol Descriptor	This describes the designation of transmission protocol and the location information of application which depends on transmission protocol.
MH-Simple application location Descriptor	This describes where to acquire application in detail.
MH-Application boundary authority setting Descriptor	This describes setting of application boundary, and setting of authority for broadcasting resource access in each area (URL).
MH-Starting priority information Descriptor	This describes starting priority of application.
MH-Cache information Descriptor	This describes the information of cache control which caches and keeps resource composing application.
MH-Probabilistic application delay Descriptor	This describes setting delay which makes timing for application control late by probability.

Descriptor name	Summary of function
Link destination PU Descriptor	This describes the information of presentation unit for link destination.
Lock cache designation Descriptor	This describes the designation of objective file to be cashed and locked.
Unlock cache designation Descriptor	This describes the designation of file to be unlocked.
MH-Download protection Descriptor*2	This describes the location information and the transmission information of MMTP packet which transmits DCM and DMM.
Application service Descriptor	This describes the entry information of application related to service and so on.
MPU node Descriptor	This represents that concerned MPU corresponds to directory node provided in the data directory management table.
PU configuration Descriptor	This represents the list of MPU which presentation unit is composed of.
MH-Layered coding Descriptor	This describes the information for identifying hierarchically encoded video stream component.
Content copy control Descriptor	This represents the control information or the maximum transmission rate about digital copy of content.
Content use control Descriptor	This describes the control information about storage and output of content.
Emergency news Descriptor	This represents emergency flash news related to security and safety (earthquake early warning, a news bulletin, prompt news superimposition) is on air.
MH-CA contract information Descriptor*2	This describes the information to confirm that service or event can be reserved.
MH-CA service Descriptor*2	This represents the program channel of broadcaster which operates automatic display message, and describes display control information of concerned message.
MH-External application control Descriptor	This describes the authority of access to broadcasting resource which is given to external application.
MH-Video record and reproduction application Descriptor	This represents the application which starts according to the reproduction of recorded content.
MH-Simple video record and reproduction application Descriptor	This represents where to acquire application in detail which starts according to the reproduction of recorded content.
MH-Application Validity term Descriptor	This represents the term of validity of application which starts at the reproduction of recorded content.
Related broadcaster Descriptor	This represents the relationship with broadcaster of other network in order to share NVRAM.
Multimedia service information Descriptor	This describes detailed information about each content of multimedia service.
Descriptor which is prepared by broadcasters	registered and open

*1 : Descriptor which is provided in Notification
*2 : Descriptor which is provided in ARIB STD-B61

Table 4-10　Assignment of descriptor tag of MMT-SI

Descriptor tag value	Descriptor name
0x0000	CRI Descriptor*2
0x0001	MPU Time stamp Descriptor*1
0x0002	Dependency relationship Descriptor *1
0x0003	GFDT Descriptor*2
0x0004 – 0x3FFF	reserved for ISO/IEC (8-bit length descriptor)
0x4000 – 0x6FFF	reserved for ISO/IEC (16-bit length descriptor)
0x7000 – 0x7FFF	reserved for ISO/IEC (32-bit length descriptor)
0x8000	Asset group Descriptor
0x8001	Event package Descriptor
0x8002	Background color designation Descriptor
0x8003	MPU presentation area designation Descriptor
0x8004	Access control Descriptor*1
0x8005	Scramble system Descriptor*1
0x8006	Message certification system Descriptor
0x8007	Emergency information Descriptor (MH)*1
0x8008	MH-MPEG-4 audio Descriptor
0x8009	MH-MPEG-4 audio extension Descriptor
0x800A	MH-HEVC video Descriptor
0x800B	Reserved (those of which descriptor length field is 8 bits) (provided by the Ministry or private standardization organization)
0x800C	MH-Event group Descriptor
0x800D	MH-Service list Descriptor
0x800E – 0x800F	Reserved (those of which descriptor length field is 8 bits) (provided by the Ministry or private standardization organization)
0x8010	Video component Descriptor
0x8011	MH-Stream identification Descriptor
0x8012	MH-Content Descriptor
0x8013	MH-Parental rate Descriptor
0x8014	MH-Audio component Descriptor
0x8015	MH-Object area Descriptor
0x8016	MH-Series Descriptor
0x8017	MH-SI transmission parameter Descriptor
0x8018	MH-Broadcaster name Descriptor
0x8019	MH-Service Descriptor
0x801A	IP data flow Descriptor
0x801B	MH-CA starting Descriptor
0x801C	MH-Type Descriptor
0x801D	MH-Info Descriptor
0x801E	MH-Expire Descriptor
0x801F	MH-CompressionType Descriptor
0x8020	MH-Data coding system Descriptor
0x8021	UTC-NPT reference Descriptor

Descriptor tag value	Descriptor name
0x8022	Reserved (those of which descriptor length field is 8 bits) (provided by the Ministry or private standardization organization)
0x8023	MH-Local time offset Descriptor
0x8024	MH-Component group Descriptor
0x8025	MH-Logo transmission Descriptor
0x8026	MPU Extension time stamp Descriptor
0x8027	MPU download content Descriptor
0x8028	MH-Network download content Descriptor
0x8029	MH-Application Descriptor
0x802A	MH-Transmission protocol Descriptor
0x802B	MH-Simple application location Descriptor
0x802C	MH-Application boundary authority setting Descriptor
0x802D	MH-Starting priority information Descriptor
0x802E	MH-Cache information Descriptor
0x802F	MH-Probabilistic application delay Descriptor
0x8030	Link destination PU Descriptor
0x8031	Lock cache designation Descriptor
0x8032	Unlock cache designation Descriptor
0x8033	MH-Download protection Descriptor[*3]
0x8034	Application service Descriptor
0x8035	MPU node Descriptor
0x8036	PU configuration Descriptor
0x8037	MH-Layered coding Descriptor
0x8038	Content copy control Descriptor
0x8039	Content usage control Descriptor
0x803A	MH-External application control Descriptor
0x803B	MH-Video recording and reproduction application Descriptor
0x803C	MH-Simple video recording and reproduction application location Descriptor
0x803D	MH-Application valid term Descriptor
0x803E	Related broadcaster Descriptor
0x803F	Multimedia service information Descriptor
0x8040	Emergency news Descriptor
0x8041	MH-CA contract information Descriptor[*3]
0x8042	MH-CA service Descriptor[*3]
0x8043 – 0xEBFF	Reserved (those of which descriptor length field is 8 bits) (provided by the Ministry or private standardization organization)
0xEC00 – 0xEFFF	Descriptor which is prepared by broadcasters (those of which descriptor length field is 8 bits)
0xF000	MH-Link Descriptor
0xF001	MH-Short format event Descriptor
0xF002	MH-Extension format event Descriptor
0xF003	Event message Descriptor

Descriptor tag value	Descriptor name
0xF004 – 0xFBFF	Reserved (those of which descriptor length field is 16 bits) (provided by the Ministry or private standardization organization)
0xFC00 – 0xFFFF	Descriptor which is prepared by broadcasters (those of which descriptor length field is 16 bits)

*1 : according to Notification
*2 : Descriptor which is provided in MMT Standard but is not used in this standard
*3 : Descriptor which is provided in ARIB STD-B61

4.3 Transmission of MMT-SI

Table 4-11 shows the value of Packet ID of MMTP Packet which transmits the message shown in Table 4-5.

The value of Packet ID of MMTP Packet which transmits message that broadcasters set can be provided in a scope that does not prevent transmitting the signal provided in Ordinance and Notification and the signal provided by ARIB. The value of Packet ID shall be registered and open as broadcaster signal. The assignment of Packet ID is shown in Table 4-12.

Table 4-11 Assignment of Packet ID of MMTP transmitting message

Message	Packet ID
PA message*	0x0000 or indirect designation by PLT
CA message*	0x0001
M2 section message which stores ECM*	Indirect designation by MPT
M2 section message which stores EMM*	Indirect designation by CAT
M2 section message which stores DCM*	Indirect designation by MPT
M2 section message which stores DMM*	Indirect designation by MH-SDTT
M2 section message which stores MH-EIT*	0x8000
M2 section message which stores MH-AIT*	0x8001or indirect designation by MPT
M2 section message which stores MH-BIT*	0x8002
M2 section message which stores MH-SDTT*	0x8003
M2 section message which stores MH-SDT*	0x8004
M2 short section message which stores MH-TOT	0x8005
M2 section message which stores MH-CDT	0x8006
Data transmission message	0x8007 or indirect designation by MPT
M2 section message which stores EMT	Indirect designation by MPT

* : according to Notification

Table 4-12 Assignment of Packet ID of MMTP

Packet ID	Meaning of Packet ID
0x0000	PA message*
0x0001	CA message*
0x0002	AL-FEC message
0x0003 – 0x00FF	Undefined
0x0100 – 0x7FFF	Provided by the Ministry or private standardization organization (region which can be assigned except for control message)
0x8000	M2 section message (MH-EIT is stored)
0x8001	M2 section message (MH-AIT is stored)
0x8002	M2 section message (MH-BIT is stored)
0x8003	M2 section message (H-SDTT is stored)
0x8004	M2 section message (MH-SDT is stored)
0x8005	M2short section message (MH-TOT is stored)
0x8006	M2 section message (MH-CDT is stored)
0x8007	Data transmission message
0x8008 – 0x8FFF	Reserved (provided by the Ministry or private standardization organization)
0x9000 – 0xFFFF	prepared by broadcasters

* : according to Notification

Chapter 5: Control Information of TLV Multiplex System

5.1 Summary of Control Information

The control information of TLV packet (TLV-SI) provides the information for the receiver to release multiplexed IP packet in broadcasting channel. TLV-SI consists of the table in the following, and the table is transmitted by the section format.

(1) Network information table for TLV (TLV-NIT):

TLV-NIT provides information about physical network.

(2) Address Map Table (AMT)

AMT provides the information which connects the service identifier by which broadcasting program number is identified with IP packet.

Also, in this standard, all “reserved” bits and “reserved_future_use” bits shall be ‘1’, unless they are defined in the other section.

5.2 Table

5.2.1 Definition of Table

5.2.1.1 Network Information Table for TLV

[Note] Network Information Table for TLV is also provided in Notification.

TLV-NIT represents the information about physical configuration of TLV stream conveyed by network and the characteristics of network itself. It is possible to define uniquely each TLV stream by combining original network identification and TLV stream identification in the whole scope of this standard.

A proper network identification is assigned to network, and it functions as the proper identification code of the network. The value of network identification is based on the provisions by the standardization organization. Also, the value of TLV stream identification can be selected independently by the broadcasters. In case that TLV-NIT is transmitted in the network where TLV stream was generated, the network identification is equivalent to the original network identification. For TLV-NIT, the group of sections which have the same table identification, the same network identification and version number is considered as a sub-table.

The configuration of network information table for TLV is shown in Table 5-1.

Table 5-1 Network Information table for TLV

Data structure	Number of bit	Data notation
TLV_Network_Information_Table () {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'1'	1	bslbf
'11'	2	bslbf
section_length	12	uimsbf
network_id	16	uimsbf
'11'	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
reserved_future_use	4	bslbf
network_descriptors_length	12	bslbf
for(i=0; i<N; i++) {		
descriptor ()		
}		
reserved_future_use	4	bslbf
TLV_stream_loop_length	12	uimsbf
for(i=0; i<N; i++) {		
tlv_stream_id	16	uimsbf
original_network_id	16	uimsbf
reserved_future_use	4	bslbf
tlv_stream_descriptors_length	12	uimsbf
for(j=0; j<N; j++) {		
descriptor ()		
}		
}		
CRC_32	32	rpchof
}		

The meaning of network information table for TLV:

table_id (table identification): according to Table 4-2

section_syntax_indicator (section syntax indicator): This shall be '1' which represents extension format.

section_length (section length): Two bits of head shall always be '00'. This specifies the number of Bytes from immediate after section length field to the end of section involving CRC. In order that the length of all sections does not exceed 1024 Bytes, the length of section must not exceed 1021.

network_id (network identification): This takes a role of label by which the delivery system represented by TLV-NIT is identified by discriminating the other delivery system.

version_number (version number): This shall be the area to write the version number of the table. In case that information in the table has been changed, one is added. If the value becomes 31, it will return to 0 next time.

current_next_indicator (current next indicator): In case of '1', this represents that the table is valid now. In case of '0', this represents that the table transmitted is not applied yet, and it will be valid next.

section_number (section number): This represents the section number. Section number in the first section in sub-table is 0x00. Every time the section which has the same table identification and network identification is added, one is added to the section number.

last_section_number (last section number): This specifies the number of last section in sub-table which the section belongs to (that is, the section which has maximum section number).

network_descriptors_length (network descriptors length): First two bits shall be '00', and the remaining 10 bits shall be the area where the number of all Bytes of successive descriptors are filled in.

tlv_stream_loop_length (TLV stream loop length): First two bits shall be '00', and the remaining 10 bits shall be the area where the number of all Bytes of TLV stream loop which ends immediately before first Byte of CRC_32.

tlv_stream_id (TLV stream identification): This represents the identification number of concerned TLV stream.

original_network_id (original network identification): This represents the identification number of original network of concerned TLV stream.

TLV_stream_descriptors_length (TLV stream descriptors length): This represents the Byte length of all descriptors of concerned TLV stream immediately after this area. But first two bits shall be '00'.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.222.0.

5.2.1.2 Address Map Table

[Note] Address Map Table is also provided in Notification.

AMT composes each service transmitted in the network. It provides the list of multicast group by IP packet. The configuration of AMT is shown in Table 5-2.

Table 5-2 Address map table

Data structure	Number of bit	Data notation
Address_Map_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'1'	1	bslbf
'11'	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
'11'	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
num_of_service_id	10	uimsbf
reserved_future_use	6	bslbf
for (i=0; i<num_of_service_id ; i++) {		
service_id	16	uimsbf
ip_version	1	bslbf
reserved_future_use	5	bslbf
service_loop_length	10	uimsbf
if (ip_version == '0') { /*IPv4*/		
src_address_32	32	bslbf
src_address_mask_32	8	uimsbf
dst_address_32	32	bslbf
dst_address_mask_32	8	uimsbf
}		
else if (ip_version == '1') { /*IPv6*/		
src_address_128	128	bslbf
src_address_mask_128	8	uimsbf
dst_address_128	128	bslbf
dst_address_mask_128	8	uimsbf
}		
for (j=0; i<N; j++) {		
private_data_byte	8	bslbf
}		
}		
CRC_32	32	rpchof
}		

The meaning of Address Map Table:

table_id (table identification): This shall be 0xFE which represents identifying table by the value of table identification extension. (Refer to Table 4-2.)

section_syntax_indicator (section syntax indicator): This shall be '1' which represents the extension format.

section_length (section length): This specifies the number of Bytes from immediately after section length field to last section involving CRC_32.

table_id_extension (table identification extension): This shall be 0x0000 which presents address map table. (Refer to Table 4-2.)

version_number (version number): This shall be the area to write the version number of the table. In case that information in the table has been changed, 1 is added. If the value becomes 31, it will return to 0 next time.

current_next_indicator (current next indicator): In case of '1', this represents the table is valid now. In case of '0', this represents the transmitted table is not applied yet, and it will be valid next.

section_number (section number): This represents the section number. Section number in the first section in sub-table is 0x00. 1 is added to the section number every time the section which has the same table identification and the same table identification extension is added.

last_section_number (last section number): This specifies the number of last section of table which the section belongs to (that is, the section which has maximum section number).

num_of_service_id (number of service identification): This represents the number of service_id which is described in this address map table.

service_id (service identification): This plays a role of label to identify service. This has the same role as service identification described in service list descriptor.

ip_version (IP version): This represents the version of IP packet which is described in list, and it is coded according to Table 5-3.

Table 5-3 IP version

IP version	Version of IP packet
0	IPv4
1	IPv6

service_loop_length (service loop length): This represents Byte length from immediately after this field to just before next service identification field.

src_address_32 (source IPv4 address): This describes sender IP address of IPv4 packet configuring service.

src_address_mask_32 (source IPv4 address mask): This designates the number of bits from effective head (MSB) for IP address to be designated sender IPv4 address. It does not take the value more than 32.

dst_address_32 (destination IPv4 address): This describes destination IP address of IPv4 address configuring service.

dst_address_mask_32 (destination IPv4 address mask): This designates the number of bits from effective head (MSB) for IP address to be designated destination IPv4 address. It does not take the value more than 32. Also, multicast group configuring service shall be the multicast group that agrees with both sender IPv4 address which is identified as effective by sender IPv4 address mask and destination IPv4 address which is identified as effective by destination IPv4 address mask.

src_address_128 (source IPv6 address): This describes sender IP address of IPv6 packet configuring service.

src_address_mask_128 (source IPv6 address mask): This designates the number of bits from effective head (MSB) for IP address to be designated sender IPv6 address. It does not take the value more than 128.

dst_address_128 (destination IPv6 address): This describes destination IP address of IPv6 address configuring service.

dst_address_mask_128 (destination IPv6 address mask): This designates the number of bits from effective head (MSB) for IP address to be designated destination IPv6 address. It does not take the value more than 128. Also, multicast group configuring service shall be the multicast group that agrees with both sender IPv6 address which is identified as effective by sender IPv6 address mask and destination IPv6 address which is identified as effective by destination IPv6 address mask.

private_data_byte: Data which is defined separately is stored in this area.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.222.0.

5.3 Descriptor

The descriptor to be arranged in TLV-SI is specified for the control information of TLV packet.

5.3.1 Definition of Descriptor

The data structure shown in the following is applied to all descriptors defined in this section.

descriptor_tag (descriptor tag): The descriptor tag is a field of 8 bits and identifies each descriptor. The value of the descriptor tag is defined in Table 4-4.

descriptor_length (descriptor length): This shall be an area where to write the number of data byte succeeding to this field.

5.3.1.1 Service List Descriptor

The service list descriptor provides service list by the service identification and the type of service format. The configuration of the service list descriptor is shown in Table 5-4.

Table 5-4 Service list descriptor

Data structure	Number of bit	Data notation
Service_List_Descriptor ()		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N+; i++) {		
service_id	16	uimsbf
service_type	8	uimsbf
}		
}		

The meaning of service list descriptor:

service_id (service identification): This is a field of 16 bits, and uniquely identifies the information service in the TLV stream.

service_type (service type classification): This is a field of 8 bits, and represents the kind of service according to Table 5-5.

Table 5-5 Service type

Service type	meaning
0x00	undefined
0x01	Digital TV service
0x02	Digital audio service
0x03 – 0x7F	undefined
0x80 – 0xA0	Defined by broadcasters
0xA1	Special video service
0xA2 – 0xA3	Reserved (ARIB STD-B10)
0xA4	Engineering service
0xA5 – 0xAD	Reserved (ARIB STD-B10)
0xAE – 0xBF	Undefined (area defined by standardization organization)
0xC0	Data service
0xC1	Storage type service using TLV
0xC2	Multimedia service
0xC3 – 0xFF	undefined

5.3.1.2 Satellite Delivery System Descriptor

The satellite delivery system descriptor represents the physical condition of satellite channel. The configuration of the satellite delivery system descriptor is shown in Table 5-6.

Table 5-6 Satellite delivery system descriptor

Data structure	Number of bit	Data notation
Satellite_Delivery_System_Descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
frequency	32	bslbf
orbital_position	16	bslbf
west_east_flag	1	bslbf
polarisation	2	bslbf
modulation	5	bslbf
symbol_rate	28	bslbf
FEC_inner	4	bslbf
}		

The meaning of satellite delivery descriptor:

frequency (frequency): This is composed of 32 bits field, and an eight-digit by 4 bits BCD code represents frequency. For the satellite delivery system descriptor, frequency is coded in GHz unit whose fourth digit and after is decimal. (for example: 012.73300GHz)

orbital_position (orbit): This is composed of 16 bits field, and a four-digit by 4 bits BCD code, the fourth digit is decimal in the degree unit, and represents the orbital position. (for example: 144.0 degree)

west_east_flag (west longitude and east longitude flag): This is 1 bit field, and represents whether the position of the satellite is in the east longitude or in the west longitude. '0' represents the satellite is located in the west longitude, and '0' represents in the east longitude.

polarization (polarization): This is composed of 2 bits field, and represents the polarization of the transmission signal according to Table 5-7. First bit specifies linearly polarized wave or circularly polarized wave.

Table 5-7 Polarization

polarization	description
00	horizontal
01	vertical
10	Counter-clockwise rotation
11	Clockwise rotation

modulation (modulation): This is composed of 5 bits field, and represents the modulation system used in the satellite delivery system, according to Table 5-8.

Table 5-8 Modulation system of satellite

Modulation system bit 4 3210	description
0 0000	undefined
0 0001	QPSK
0 1000	Wide band satellite digital broadcasting system (Refer to TMCC signal)
0 1001	2.6 GHz band satellite digital audio broadcasting system (Refer to Pilot channel)
0 1010	Advanced narrow band CS digital broadcasting system (Refer to physical layer header and base band header)
0 1011	Advanced wide band satellite digital broadcasting system (Refer to TMCC signal)
0 0010 – 0 0111 0 1100 – 1 1111	Reserved for use in the future

symbol_rate (symbol rate): This is composed of 28 bits field, a 7-digits by 4 bits BCD code, the fourth digit is decimal in Msymbol/s unit, and represents the value of symbol rate. (for example: 021.0960)

FEC_inner (FEC (inner-code)): This is composed of 4 bits field, and represents the inner-code according to Table 5-9.

Table 5-9 FEC (inner code)

FEC (inner-code) bit 3210	description
0000	undefined
0001	Coding rate 1/2
0010	Coding rate 2/3
0011	Coding rate 3/4
0100	Coding rate 5/6
0101	Coding rate 7/8
1000	Wide band satellite digital broadcasting system (Refer to TMCC signal)
1001	2.6 GHz band satellite digital audio broadcasting system (Refer to pilot channel)
1010	Advanced narrow band CS digital broadcasting system (Refer to physical layer header)
1011	Advanced wide band satellite digital broadcasting system (Refer to TMCC signal)
1111	Without inner-code
0110 – 0111 1100 – 1110	Reserved for use in the future

5.3.1.3 System Management Descriptor

[Note] This item is provided in Notification.

The system management descriptor is used for discriminating between broadcasting and non-broadcasting. The configuration of system management descriptor is shown in Table 5-10.

Table 5-10 System management descriptor

Data structure	Number of bit	Data notation
System_Management_Descriptor(){		
descriptor_tag	8	Uimsbf
descriptor_length	8	Uimsbf
system_management_id	16	Uimsbf
for (i=0; i<N; i++) {		
additional_identification_info	8	Uimsbf
}		
}		

The meaning of system management descriptor:

system_management_id (System management identification): This is composed of 16 bits field, and the configuration is shown in table 5-11.

Table 5-11 Configuration of system management identification

Data structure	Number of bit	Data notation
system_management_id () {		
broadcasting_flag	2	uimsbf
broadcasting_identifier	6	uimsbf
additional_broadcasting_identification	8	uimsbf
}		

The meaning of system management identification:

broadcasting_flag (Broadcasting/Non-broadcasting classification): This is composed of 2 bits field, and represents Broadcasting or Non-broadcasting according to Table 5-12.

Table 5-12 Broadcasting/non-broadcasting

value	meaning
00	Broadcasting
01 10	Non-Broadcasting
11	undefined

broadcasting_identifier (kind of standard system of broadcasting): This is composed of 6 bits field, and represents the standard system of broadcasting, according to Table 5-13.

Table 5-13 Kind of standard system of broadcasting

value	meaning
000000	undefined
000001	Standard system that is specified as satellite digital broadcasting by narrow band transmission system which uses 27MHz bandwidth in the frequency range from 12.2 to 12.75GHz
000010	Standard system that is specified as satellite digital broadcasting by wide band transmission system which uses 34.5MHz bandwidth in the frequency range from 11.7 to 12.2GHz
000011	Standard system that is specified as digital terrestrial television broadcasting
000100	Standard system that is specified as satellite digital broadcasting by wide band transmission system which uses 34.5MHz bandwidth in the frequency range from 12.2 to 12.75GHz
000101	Standard system that is specified as digital terrestrial sound broadcasting
000110	Very high frequency broadcasting that satellite broadcasting station and broadcasting station which use radio wave whose frequency is more than 2630MHz and less than or equal to 2655MHz.
000111	Standard system that is specified as satellite digital broadcasting by advanced narrow band transmission system which uses 27MHz bandwidth in the frequency range from 12.2 to 12.75GHz
001000	Standard system that is specified as satellite digital broadcasting by advanced wide band transmission system which uses 34.5MHz bandwidth in the frequency range from 11.7 to 12.2GHz
001001	Standard system that is specified as satellite digital broadcasting by advanced wide band transmission system which uses 34.5MHz bandwidth in the frequency range from 12.2 to 12.75GHz
001010	Standard system that is specified as the broadcasting by connected segment transmission system among multimedia broadcasting which broadcasting station provides by using radio wave whose frequency is from 207.5MHz to 222MHz
001011 – 111111	undefined

additional_broadcasting_identification (detailed identification): This is composed of 8 bits field, and is provided in the operational guidelines for broadcasters.

additional_identification_info (additional identification information): This is composed of 8 bits field, and is used for extension of the identification number of the system management.

5.3.1.4 Network Name Descriptor

The network name descriptor describes the network name by character code. The configuration of the network name descriptor is shown in Table 5-14.

Table 5-14 Network name descriptor

Data structure	Number of bit	Data notation
Network_Name_Descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++){		
char	8	uimsbf
}		
}		

The meaning of network name descriptor:

char (character code): This is a field of 8 bits. In a series of character code field, the name of the delivery system which is notified by TLV-NIT is described.

5.3.1.5 Remote Control Key Descriptor

Remote control key descriptor sets up services which are assigned to one-push selection button of remote controller for the receiver. Remote control key descriptor is arranged in the first loop of TLV-NIT. The configuration of remote control key descriptor is shown in Table 5-15.

Table 5-15 Remote control key descriptor

Data structure	Number of bit	Data notation
Remote_Control_Key_Descriptor () {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
num_of_ remote_control_key_id	8	uimsbf
for (i=0; i< num_of_ remote_control_key_id; i++) {		
remote_control_key_id	8	uimsbf
service_id	16	uimsbf
reserved	16	bslbf
}		
}		

The meaning of remote control key descriptor:

num_of_remote_control_key_id (number of remote control key): This is composed of 8 bits field, and represents the number of keys for remote controller to which service identification is assigned by this descriptor.

remote_control_key_id (remote control key identification): This is composed of 8 bits field, and represents recommended value of the button number of remote controller to which the service designated by successive service identification is assigned.

service_id (service identification): This is composed of 16 bits field, and represents the service which is assigned to the remote control key identification.

Chapter 6: Coded Signal of MMT

6.1 Summary of Coded Signal

There are Media Fragment Unit (MFU), Media Processing Unit (MPU), MMTP Payload, and MMTP Packet as elements composing coded signals in MMT. The summary of coded signal in MMT is shown in Fig. 6-1.

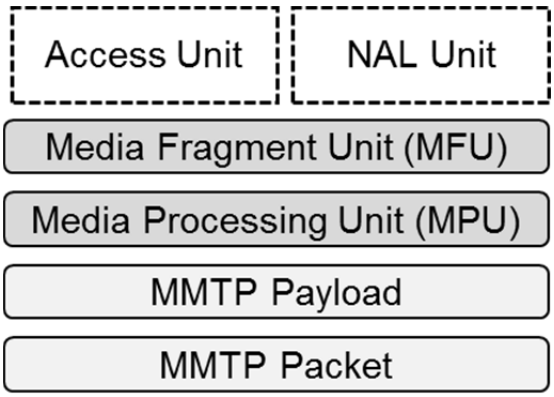


Fig. 6-1 Summary of coded signal in MMT

6.2 Summary of MFU/MPU

For the processing of video and audio signals, MPU is the unit of processing. MPU involves more than one access unit, and MPU alone becomes a unit by which video and audio decoding can be processed. The size of MPU is arbitrary, and it can involve arbitrary number of access units. For video signal which is decoded by using inter-frame prediction, MPU needs to be the unit of GOP. The general configuration of MPU is shown in Fig. 6-2.

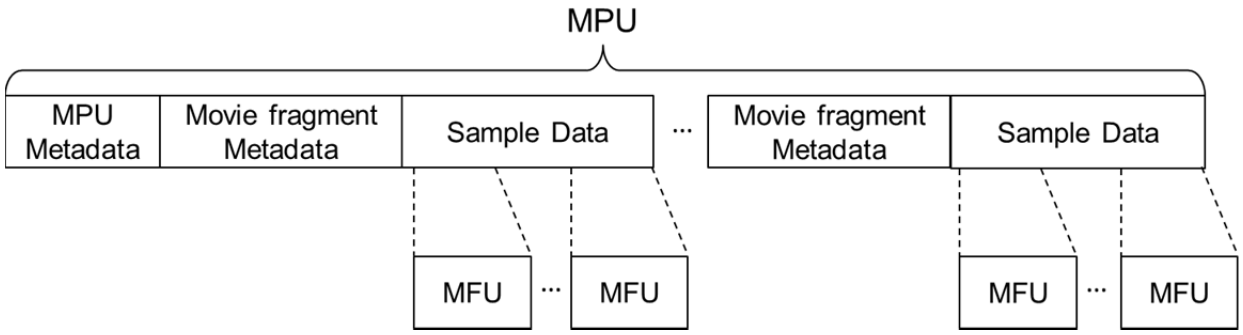


Fig. 6-2 General configuration of MPU

MPU consists of MPU metadata which involves the information regarding configuration of whole MPU, movie fragment metadata which involves information regarding coded media data, and sample data which is coded media data. As one MPU can consist of one or more than one movie fragments, it would be possible that there are plural movie fragment data and sample data. For MPU, sequence number is added to each MPU which belongs to the same asset. By using the asset ID which identifies asset and the sequence number of MPU, MPU can be distinguished from the other MPU.

MFU is a smaller unit than MPU, and MFU can be taken from sample data. As a configuration method of MFU, MFU can be configured by the unit of NAL unit or by the unit of

access unit. By configuring MFU being aware of media, and by transmitting on the unit of MFU, an error propagation such as a packet loss can be suppressed when channel quality happens to deteriorate.

For media of video component and audio component, presentation time and decoding time can be designated by the unit of MPU or by the unit of access unit. Based on UTC, each transmitter designates these time by using common time-axis. So necessary media component can be represented in synchronization without depending on the difference of channel for broadcasting and communication or the difference of transmitters.

There are two ways to configure MMTP Payload by MPU and MFU. One method is that by NAL unit and access unit of video signal and audio signal which is output by the encoder, after MPU with general format shown in Fig. 6-2 is configured, the MPU is divided and MMTP Payload is configured. Another method is that omitting the process of configuring MPU with general format shown in Fig. 6-2, MMTP Payload is directly configured by NAL unit and access unit. For shortening delay, the latter method shall be used in broadcasting. Also, codec information, etc. which is prepared in MPU metadata and movie fragment metadata is provided as control information. So, MPU metadata and movie fragment metadata shall not be transmitted.

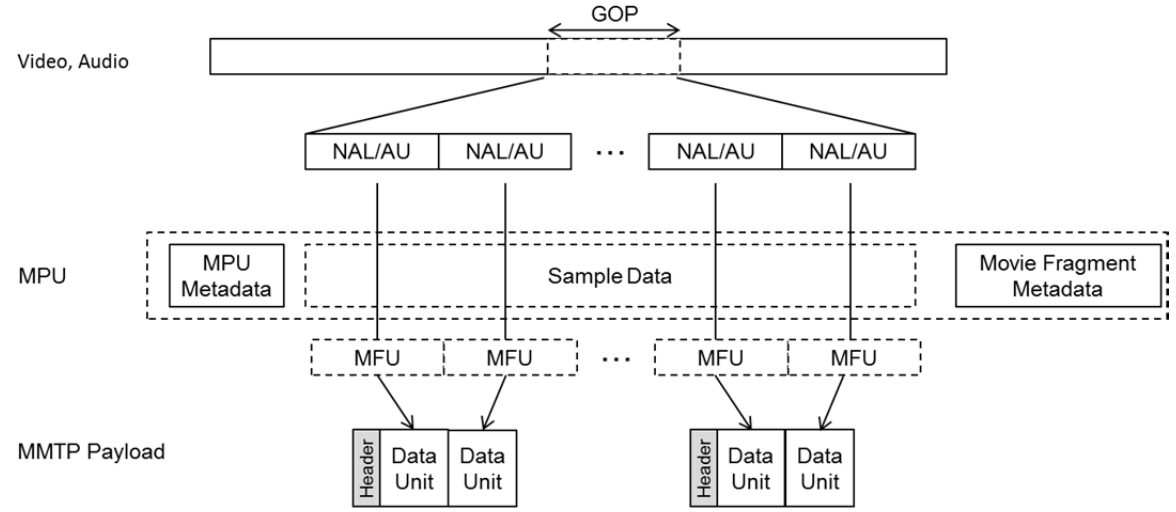
6.3 MMTP Payload

6.3.1 Summary of MMTP Payload

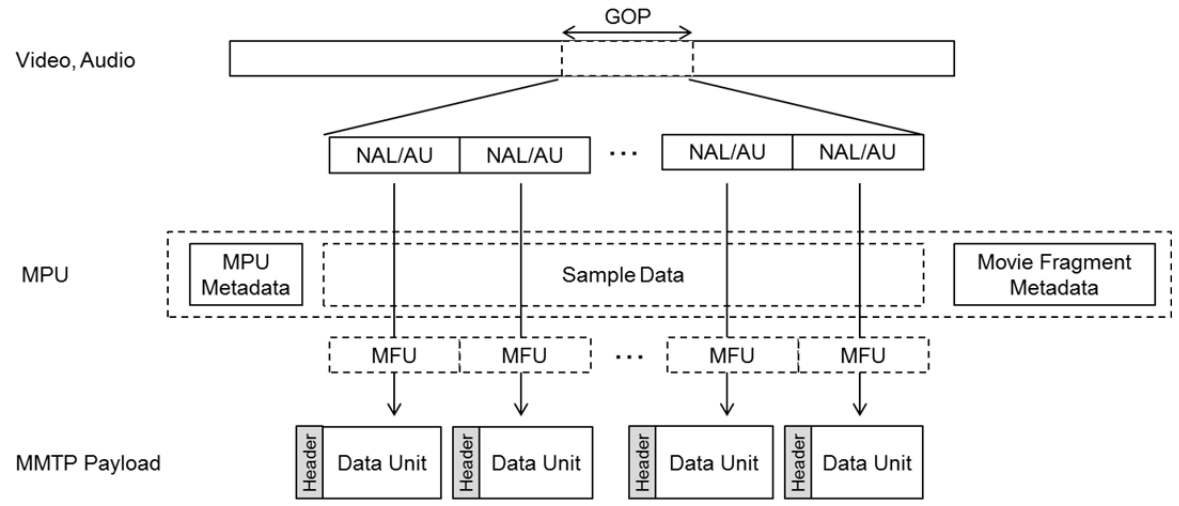
The control information of MFU and MMT is transmitted by MMTP packet. MMTP packet consists of header part and payload part, and the payload part is called MMTP Payload.

When the size of MFU to be transmitted is small, MMTP Payload can be configured by collecting plural MFU of the same kind. On the other hand, when the size of thing to be transmitted is big and it cannot be transmitted by one IP packet, it is divided and configured to plural payloads.

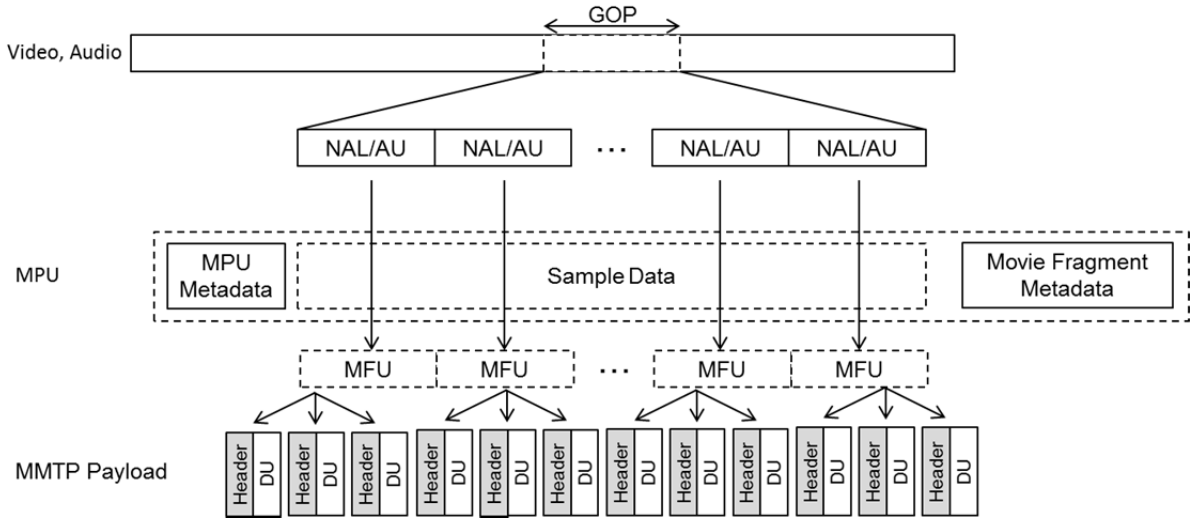
Outline from configuring MPU by video and audio signal to storing it in MMTP Payload is shown in Fig. 6-3. Fig. 6-3 (a) shows an example that the size of MFU is smaller enough than MTU, and plural MFUs are stored in one payload. Fig. 6-3 (b) shows an example that the size of MFU is smaller than MTU, and each MFU is stored in one payload. Moreover Fig. 6-3 (a) shows an example that the size of MFU is big, so it is stored in plural payloads after dividing.



(a) An example that plural MFUs are stored in one MMTP payload



(b) An example that one MFU is stored in one MMTP payload



(c) An example that one MFU is divided into plural MMTP payloads

Fig. 6-3 Summary of configuring MMTP payload from video and audio signal

As the offset information of MFU, the information representing the condition of division of MFU, etc. are stored in MMTP Payload, the head of access unit is detected based on the information of MMTP Payload in the receiver.

6.3.2 Configuration of MMTP Payload

The configuration of MMTP Payload is shown in Table 6-1. A part of the configuration of MMTP Payload is also provided in Notification.

Table 6-1 Configuration of MMTP payload

Data structure	Number of bit	Data notation
MMTP_payload () {		
if (payload_type == 0x00) {		
/* media aware fragment MPU */		
payload_length	16	uimsbf
fragment_type	4	uimsbf
timed_flag	1	bslbf
fragmentation_indicator	2	bslbf
aggregation_flag	1	bslbf
fragment_counter	8	uimsbf
MPU_sequence_number	32	uimsbf
if (fragment_type == 2) { //MFU		
if (timed_flag == 1) { //timed data		
if (aggregation_flag == 0) {		
movie_fragment_sequence_number	32	uimsbf
sample_number	32	uimsbf
offset	32	uimsbf
priority	8	uimsbf
dependency_counter	8	uimsbf
for (j=0; j<M; j++) {		
MFU_data_byte	8	bslbf
}		
} else {		
for (i=0; i<N; i++) {		
data_unit_length	16	uimsbf
movie_fragment_sequence_number	32	uimsbf
sample_number	32	uimsbf
offset	32	uimsbf
priority	8	uimsbf
dependency_counter	8	uimsbf
for (j=0; j<M; j++) {		
MFU_data_byte	8	bslbf
}		
}		
}		
} else { //non-timed data		
if(aggregation_flag == 0) {		

item_id	32	uimsbf
for (j=0; j<M; j++) {		
MFU_data_byte	8	bslbf
}		
} else {		
for (i=0; i<N; i++) {		
data_unit_length	16	uimsbf
item_id	32	uimsbf
for (j=0; j<M; j++) {		
MFU_data_byte	8	bslbf
}		
}		
}		
}		
}		
}		
if (payload_type == 0x02) {		
/* signaling message */		
fragmentation_indicator	2	bslbf
reserved	4	bslbf
length_extension_flag	1	bslbf
aggregation_flag	1	bslbf
fragment_counter	8	uimsbf
if (aggregation_flag == 0) {		
for (j=0; j<M; j++) {		
message_byte	8	bslbf
}		
} else {		
for (i=0; i<N; i++) {		
if (length_extension_flag ==1) {		
message_length	32	uimsbf
} else {		
message_length	16	uimsbf
}		
for (j=0; j<M; j++) {		
message_byte	8	bslbf
}		
}		
}		
}		
}		

The meaning of MMTP payload:

payload_length (payload length): This represents the number of Bytes from immediately after this field to the last of MMTP Payload.

fragment_type (fragment type): This represents the kind of fragment of information which is stored in MMTP Payload, and coded according to Table 6-2.

Table 6-2 Fragment type

Value of fragment type	Meaning of fragment type
0*1	MPU metadata. ftyp, mmpu, moov, meta box are involved.
1*1	Movie fragment metadata. moof box and, mdat box excluding media data are involved.
2	MFU. Sample or sub-sample (media data with time), or items (media data without time) are involved.
3 – 15	provided by the Ministry or private standardization organization.

*1: These are provided in MMT Standard, but are not used in this standard.

timed_flag (time data flag): When the data which MMTP Payload stores is the data which designates presentation time, this shall be '1', and when it is not the data which does not designate presentation time, this shall be '0'.

fragmentation_indicator (fragmentation indicator): This represents the condition of data fragmentation to be stored in MMTP Payload, and coded according to Table 6-3.

Table 6-3 Fragmentation indicator

Fragmentation indicator	Meaning of fragmentation indicator
00	This involves one or more than one data in perfect form.
01	This involves header part of divided data.
10	This involves a part of divided data which is neither header part nor last part.
11	This involves last part of divided data.

aggregation_flag (aggregation flag): This shall be '1' in case that more than one data is stored in MMTP Payload, and shall be '0' in case that only one data is stored in MMTP Payload.

fragment_counter (fragmentation number counter): When data is divided, this indicates the number of fragmented data which is after the part where this MMTP Payload stores. When the number of fragmentation is more than 255, this indicates the remainder with which the number of fragmented data divided by 255. Also, when the aggregation flag is '1', this field shall be '0'.

MPU_sequence_number (MPU sequence number): When MPU metadata, movie fragment metadata, MFU are stored, this indicates the sequence number of MPU which they belong to.

movie_fragment_sequence_number (movie fragment sequence number): This indicates the sequence number of the movie fragment which this MFU belongs to.

sample_number (sample number): This indicates the sample number of this MFU.

offset (MFU offset): This represents MFU offset by the unit of Byte in the sample which this MFU belongs to.

priority (MFU priority): This represents the relative priority of MFU in MPU which this MFU belongs to. This represents that if the number of MFU priority is bigger, the MFU is more important than the MFU whose number of MFU priority is small.

dependency_counter (MFU dependency number): Decoding processing depends on this MFU. Namely, this represents the number of MFU which cannot be decoded unless this MFU is decoded.

MFU_data_byte (MFU data): This represents the data byte of NAL unit, access unit or item.

data_unit_length (MFU length): This represents the size by the unit of Byte from immediately after this field to the last of one MFU data.

item_id (item identification): This represents ID which identifies item.

length_extension_flag (length information extension flag): This shall be '1' when the message data length field which represents the size of message is 32 bits, and shall be '0' when 16 bits.

message_length (message data length): This represents the size of one succeeding message from immediately after this field by the unit of Byte.

message_byte (message data): This represents the data byte of control information.

6.4 MMTP Packet

6.4.1 Summary of MMTP Packet

MMTP Payload is transmitted by one MMTP Packet. One MMTP Packet neither conveys plural MMTP Payloads, nor one MMTP Payload extends over plural MMTP Packets.

Also, MMTP Packet is transmitted by one IP packet. One IP packet neither conveys plural MMTP payloads, nor one MMTP Payload extends over plural IP packets.

6.4.1.1 Configuration of MMTP Packet

The configuration of MMTP Packet is shown in Table 6-4. The configuration of MMTP Packet is also provided in Notification.

Table 6-4 Configuration of MMTP packet

Data structure	Number of bit	Data notation
MMTP_packet () {		
version	2	uimsbf
packet_counter_flag	1	bslbf
FEC_type	2	uimsbf
reserved	1	bslbf
extension_flag	1	bslbf
RAP_flag	1	bslbf
reserved	2	bslbf
payload_type	6	uimsbf
packet_id	16	uimsbf
timestamp	32	uimsbf
packet_sequence_number	32	uimsbf
if (packet_counter_flag == 1) {		
packet_counter	32	uimsbf
}		
if (extension_flag == 1) {		
extension_type	16	uimsbf
extension_length	16	uimsbf
for (i=0; i<N; i++) {		
header_extension_byte	8	bslbf
}		
}		
MMTP_payload ()		
}		

The meaning of MMTP Packet:

version (version): This represents the version number of MMT protocol. When this is according to MMT Standard version 1, this field shall be '00'.

packet_counter_flag (packet counter flag): When the packet counter exists, this shall be '1', and when it does not exist, this shall be '0'.

FEC_type (FEC type): Information about AL-FEC of this MMTP Packet is coded according to Table 6-5.

Table 6-5 FEC type

Value of FEC type	Meaning of FEC type
0	MMTP Packet which is not protected by AL-FEC
1	Source packet, among MMTP Packets which are protected by AL-FEC
2	Repair packet, among MMTP Packets which are protected by AL-FEC
3	reserved for future use

extension_flag (extension header flag): When MMTP Packet is extended for header, this shall be '1', and when it is not extended, this shall be '0'.

RAP_flag (RAP flag): If MMTP Payload which is transmitted by this MMTP Packet involves the head of random access point, this shall be '1', otherwise this shall be '0'.

payload_type (payload type): This represents data type of MMTP Payload, and it is coded according to Table 6-6.

Table 6-6 Payload type

Value of payload type	Meaning of payload type
0x00	MPU. This involves fragment of MPU considering media.
0x01*1	Generic object. This involves general object such as a complete MPU.
0x02	This involves more than or equal to one control message.
0x03	This involves repair symbol of AL-FEC.
0x04 – 0x1F	reserved for ISO/IEC
0x20 – 0x3F	provided by the Ministry or private standardization organization

*1: This is provided in MMT Standard, but it is not used in this standard.

packet_id (packet identifier): This shall be the area for identifying the kind of data of payload. The value of packet identifier shall be the value specified in Table 4-11.

timestamp (distribution timestamp): The time when the head byte of this MMTP Packet is output from transmission entity is represented by short format NTP time stamp recorded in RFC 5905.

packet_sequence_number (packet sequence number): This represents the order of MMTP Packet having the same packet identifier. This starts from an arbitrary value.

packet_counter (packet counter): This represents the order of MMTP Packet in the same IP dataflow irrespective of the value of packet identifier. This starts from an arbitrary value.

extension_type (extension header type): This represents the kind of extension in header extension area. The value of extension header type is not provided in MMT Standard. It shall follow the assignment in table 6-7.

Table 6-7 Extension header type

Value of extension header type	Meaning of extension header type
0x0000	Multi-type header extension (This shall be the multi-type header extension whose configuration is shown in Table 6-8.)
0x0001 – 0xFFFF	provided by the Ministry or private standardization organization

extension_length (extension header length): This represents the size from immediately after this field to the last of extension header area by the unit of Byte.

header_extension_byte (extension header area): This represents data byte for header extension. When the value of extension header type is 0x0000, this shall be the structure shown in Table 6-8.

Table 6-8 Configuration of multi type header extension

Data structure	Number of bit	Data notation
Header_extension_byte {		
for (i=0; i<N; i++) {		
hdr_ext_end_flag	1	bslbf
hdr_ext_type	15	uimsbf
hdr_ext_length	16	uimsbf
for (j=0; j<M; j++) {		
hdr_ext_byte	8	bslbf
}		
}		
}		

The meaning of multi type header extension:

hdr_ext_end_flag (multi-type header extension end flag): If multi-type header extension immediately after is the last of header extension, this shall be '1', otherwise '0'.

hdr_ext_type (multi extension header type): This represents the kind of extension of multi-type header extension. This shall follow the assignment in Table 6-9.

Table 6-9 Multi extension header type

Multi extension header type	Meaning of multi extension header type
0x0000	reserved for the future
0x0001	Various kinds of information related to scramble which is specified in ARIB STD-B61 are described.
0x0002	Download ID (32 bits) recorded in Chapter 10 is described.
0x0003	Information related to file division transmission recorded in Chapter 10 is described.
0x0004 – 0x7FFF	reserved for the future

hdr_ext_length (multi extension header length): From immediately after this field, this represents the size of one extension header area immediately after (the size of hdr_ext_byte immediately after) by the unit of Byte.

hdr_ext_byte (multi extension header area): This represents the data byte for multi-type header extension.

Chapter 7: Control Information of MMT

7.1 Summary of Control Information

The control information is a transmission control signal which represents the information related to the configuration of MMT package and the broadcasting service, and is composed of three layers, 1) “message” which stores tables and descriptors, 2) “table” which has elements and attributes representing particular information, and 3) “descriptor” which represents more detailed information, as shown in Fig. 7-1.

The coding letter group for letters which are used in the control information shall be UCS (ISO/IEC 10646:2012), and the coding scheme for letters shall be UTF-8 without BOM (Byte Order Mark). For details, the provisions of ARIB STD-B62 shall be followed.

In the following, the message is provided in section 7.2, the table is provided in section 7.3, and the descriptor is provided in section 7.4.

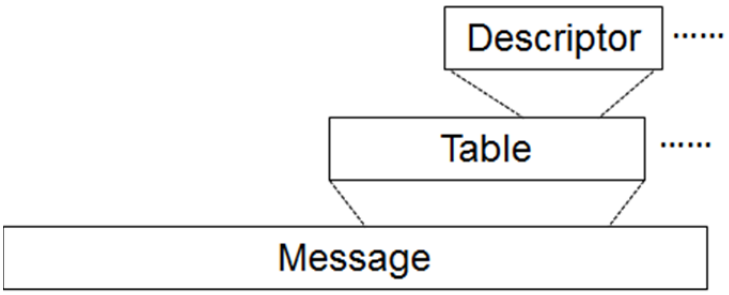


Fig. 7-1 Configuration of control information

7.2 Message

7.2.1 Summary of Message

The message can store more than or equal to one table according to its kind.

7.2.2 Transmission of Message

The message is stored in MMTP payload and transmitted by using MMTP packet. In configuring MMTP payload, the type of payload shall be 0x02 (payload which includes more than or equal to one control message). Plural messages shall not be stored in one MMTP payload, but one message shall be stored in one MMTP payload. However, when the size of message is bigger comparing with MTU, message shall be divided and stored in plural MMTP payloads, and transmitted by using plural MMTP packets.

7.2.3 Definition of Message

The data structure shown in the followings shall be applied to all messages which are defined in this section, unless there is no detailed individual description.

message_id (message identifier): Message identifier is a field of 16 bits and identifies each message. The value of message identifier is defined in Table 4-6.

version (version): This shall be the region in which the version number of message is written.

length (message length): The size from immediately after this field to the end of message payload is represented by the unit of byte.

7.2.3.1 PA Message

[Note] PA message is also provided in Notification.

PA message is used for transmitting various tables. The configuration of PA message is shown in Table 7-1.

Table 7-1 Configuration of PA message

Data structure	Number of bit	Data notation
PA_Message () {		
message_id	16	uimsbf
version	8	uimsbf
length	32	uimsbf
extension {		
number_of_tables	8	uimsbf
for (i=0; i<N; i++) {		
table_id	8	uimsbf
table_version	8	uimsbf
table_length	16	uimsbf
}		
}		
message_payload {		
for (i=0; i<N; i++) {		
table ()		
}		
}		
}		

The meaning of PA message:

number_of_tables (number of tables): This represents the number of tables which are stored in this PA message.

table_id (table identification): This represents the table identification of table which is stored in this PA message.

table_version (table version): This represents the version of table which is stored in this PA message.

table_length (table length): The size of table which is stored in this PA message is represented by the unit of byte.

table (table): This represents the table which is stored in this PA message.

7.2.3.2 M2 Section Message

[Note] M2 section message is also provided in Notification.

M2 section message is used for transmitting section extention format of MPEG-2 Systems. The configuration of M2 section message is shown in Table 7-2.

Table 7-2 Configuration of M2 section message

Data structure	Number of bit	Data notation
M2section_Message () {		
message_id	16	uimsbf
version	8	uimsbf
length	16	uimsbf
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'1'	1	bslbf
'11'	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
'11'	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for(i=0; i<N; i++) {		
signaling_data_byte	8	bslbf
}		
CRC_32	32	rpchof
}		

The meaning of M2 section message:

table_id (table identification): This shall be the region to use for identification of the table which section belongs to.

section_syntax_indicator (section syntax indicator): This shall be '1' which represents extension format.

section_length (section length): This shall be the region where to write the number of data byte which follows after the region of section length.

table_id_extension (table identification extension): This shall be the region where the table identification is extended.

version_number (version number): This shall be the region where to write the version number of table.

current_next_indicator (current next indicator): This shall be '1' when the table can be used now, and this shall be '0' when it represents that the table cannot be used now and it will be valid next.

section_number (section number): This shall be the region where to write the section number which configures the table.

last_section_number (last section number): This shall be the region where to write the last section number which configures table.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.220. The range for calculating CRC shall be from the table identification field to immediately before this field.

7.2.3.3 CA Message

[Note] CA message is also provided in Notification.

The CA message is used for transmitting the table which is used to identify the conditional access system. The configuration of CA message is shown in Table 7-3.

Table 7-3 Configuration of CA message

Data strcture	Number of bit	Data notation
CA_Message () {		
message_id	16	uimsbf
version	8	uimsbf
length	16	uimsbf
table ()		
}		

The meaning of CA message:

table (table): This represents the table which is stored in this message.

7.2.3.4 M2 Short Section Message

The M2 short section message is used for transmitting the section short format of MPEG-2 Systems. Configuration of M2 short section message is shown in Table 7-4.

Table 7-4 Configuration of M2 short section message

Data strcture	Number of bit	Data notation
M2short_Section_Message () {		
message_id	16	uimsbf
version	8	uimsbf
length	16	uimsbf
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'1'	1	bslbf
'11'	2	bslbf
section_length	12	uimsbf
for(i=0; i<N; i++) {		
signaling_data_byte	8	bslbf
}		
}		

The meaning of M2 short section message:

table_id (table identification): This shall be the region for use to identify the table which section belongs to.

section_syntax_indicator (section syntax indicator): This shall be '0' which represents short format.

section_length (section length): This shall be the region to write the number of data byte which follows from the section length region.

7.3 Table

7.3.1 Summary of Table

Table is the control information which has the elements representing specific information and the attribute, and is stored in the message and transmitted by MMTP packet.

7.3.2 Storing Table in Message

The Message in which table is stored is decided according to the table. The Message in which table is stored is shown in Table 7-5.

Table 7-5 Message stored in table

Table	Message			
	PA	M2 section	CA	M2 short section
MPT	○			
PLT	○			
LCT	○			
ECM		○		
EMM		○		
DCM		○		
DMM		○		
CAT (MH)			○	
MH-EIT		○		
MH-CDT		○		
MH-BIT		○		
MH-SDTT		○		
MH-SDT		○		
MH-TOT				○

7.3.3 Definition of Table

The data structure shown in the followings shall be applied to all the tables defined in this section, unless there is no detailed individual description.

table_id (table identifier): Table identifier is a field of 8 bits, and identifies each table. The value of table identifier is defined in Table 4-8.

version (version): This shall be the region where the version number of the table is written.

length (table length): This shall be the region where the number of data byte which follows after this field is written.

7.3.3.1 MMT Package Table (MPT)

[Note] MMT package table is also provided in Notification.

The MMT package table (MP table) gives information of which package consists, such as the list of asset and the position of asset on network, etc. Descriptors to be stored in the MMT package table shall be the descriptors which are specified in this standard. The configuration of the MP table is shown in Table 7-6.

Table 7-6 Configuration of MP table

Data strcture	Number of bit	Data notation
MMT_Package_Table () {		
table_id	8	uimsbf
version	8	uimsbf
length	16	uimsbf
reserved	6	bslbf
MPT_mode	2	bslbf
MMT_package_id_length	8	uimsbf
for (i=0; i<N; i++) {		
MMT_package_id_byte	8	bslbf
}		
MPT_descriptors_length	16	uimsbf
for (i=0; i<N; i++) {		
MPT_descriptors_byte	8	bslbf
}		
number_of_assets	8	uimsbf
for (i=0; i<N; i++) {		
identifier_type	8	uimsbf
asset_id_scheme	32	uimsbf
asset_id_length	8	uimsbf
for (j=0; j<M; j++) {		
asset_id_byte	8	uimsbf
}		
asset_type	32	char
reserved	7	bslbf
asset_clock_relation_flag	1	bslbf
location_count	8	uimsbf
for (j=0; j<M; j++) {		
MMT_general_location_info ()		
}		
asset_descriptors_length	16	uimsbf
for (j=0; j<M; j++) {		
asset_descriptors_byte	8	bslbf
}		
}		
}		

The meaning of MP table:

table_id (table identification): If this table is a completely configured MPT, it shall be 0x20. If

the configuration of one package is described by several MPT's, it shall be 0x11 – 0x1F according to subset.

MPT_mode (MPT mode): This represents the working when MPT is divided into subsets, and it shall follow the assignment in Table 7-7.

Table 7-7 MPT mode

value	Meaning of MPT mode
00	MPT is processed according to the order of subset.
01	After MPT of subset 0 is received, any subset which has the same version number can be processed.
10	MPT of subset can be processed arbitrarily.
11	reserved

MMT_package_id_length (package ID length): This represents the length of package ID byte by the unit of byte.

MMT_package_id_byte (package ID byte): This represents the package ID. Lower 16 bits of the package ID shall be the same value as the service identification for identifying service.

MPT_descriptors_length (MPT descriptors length): This represents the length of the MPT descriptor region by the unit of byte.

MPT_descriptors_byte (MPT descriptors region): This shall be the region where the descriptor of MPT is stored.

number_of_assets (number of assets): This represents the number of assets to which this table gives information.

identifier_type (identifier type): This represents ID system of MMTP packet flow. If it is ID system which represents asset ID, it shall be 0x00.

asset_id_scheme (asset ID scheme): This represents the scheme of asset ID.

asset_id_length (asset ID length): This represents the length of asset ID byte by the unit of byte.

asset_id_byte (asset ID byte): This represents asset ID.

asset_type (asset type): This represents the kind of asset according to Table 7-8.

Table 7-8 Asset type

letter	Meaning of asset type
hvc1	HEVC which is provided in ITU-T Recommendation H.265 (which includes VPS, SPS and PPS only in MPU metadata)
hev1	HEVC which is provided in ITU-T Recommendation H.265 (which includes VPS, SPS and PPS in MFU)
mp4a	ISO/IEC 14496-3 Audio
stpp	Timed text (closed-caption and superimposition)
aapp	Application
asgd	Synchronous type general-purpose data
aagd	Asynchronous type general-purpose data

asset_clock_relation_flag (clock information flag): This represents whether there is the asset clock relation flag or not. When it is '1', it shows there are a clock relation identification field and

time scale flag field. When it is '0', it shows there is neither field.

location_count (location count): This represents the number of the location information of asset.

MMT_general_location_info (location information): This represents the location information of asset.

asset_descriptors_length (asset descriptors length): This represents the length of all bytes of the following descriptors.

asset_descriptors_byte (asset descriptors region): This shall be the region where descriptors of asset are stored. Location information shall be the configuration shown in Table 7-9.

Table 7-9 Configuration of MMT_general_location_info (location information)

Data strcture	Number of bit	Data notation
MMT_general_location_info () {		
location_type	8	uimsbf
if (location_type == 0x00) {		
packet_id	16	uimsbf
}		
if (location_type == 0x01) {		
ipv4_src_addr	32	uimsbf
ipv4_dst_addr	32	uimsbf
dst_port	16	uimsbf
packet_id	16	uimsbf
}		
if (location_type == 0x02) {		
ipv6_src_addr	128	uimsbf
ipv6_dst_addr	128	uimsbf
dst_port	16	uimsbf
packet_id	16	uimsbf
}		
if (location_type == 0x03) {		
network_id	16	uimsbf
MPEG_2_transport_stream_id	16	uimsbf
reserved	3	bslbf
MPEG_2_PID	13	uimsbf
}		
if (location_type == 0x04) {		
ipv6_src_addr	128	uimsbf
ipv6_dst_addr	128	uimsbf
dst_port	16	uimsbf
reserved	3	bslbf
MPEG_2_PID	13	uimsbf
}		
if (location_type == 0x05) {		
URL_length	8	uimsbf
for (i=0; i<N; i++) {		
URL_byte	8	char

} } }		

The meaning of MMT_general_location_info (location information):

location_type (location type): This represents the kind of location information, and follows the assignment shown in Table 7-10.

Table 7-10 Location type

value	Meaning of location type
0x00	This represents MMTP packet of the same IP data flow as IP data flow in which the table including this general_location_info is transmitted.
0x01	This represents MMTP packet of IPv4 data flow.
0x02	This represents MMTP packet of IPv6 data flow.
0x03	This represents MPEG-2 TS packet of broadcasting network by MPEG-2 TS.
0x04	This represents MPEG-2 TS packet of IPv6 data flow.
0x05	This represents URL.

packet_id (packet identifier): This represents the packet ID of MMTP packet.

ipv4_src_addr (source IPv4 address): This represents the source address of IPv4 data flow.

ipv4_dst_addr (destination IPv4 address): This represents the destination address of IPv4 data flow.

dst_port (destination port number): This represents the destination port number of IP data flow.

Ipv6_src_addr (source IPv6 address): This represents the source address of IPv6 data flow.

Ipv6_dst_addr (destination IPv6 address): This represents the destination address of IPv6 data flow.

network_id (network identifier): This represents the network identification for identifying broadcasting network.

MPEG_2_transport_stream_id (transport stream identifier): This represents the transport stream identification for identifying MPEG-2 TS.

MPEG_2_PID (MPEG-2 packet identifier): This represents the packet identification for MPEG-2 TS packet.

URL_length (URL length): This represents the length of URL byte field by the unit of byte.

URL_byte (URL byte): This represents URL.

7.3.3.2 Package List Table (PLT)

The package list table shows the list of IP data flow which transmits PA message of MMT package supplied as broadcasting service, packet ID, and the list of IP data flow which transmits IP service. Descriptors to be stored in the package list table shall be the descriptors which are provided in this standard. An outline of referring to MPT of the other packages using the package list table is shown in Fig. 7-2, and the configuration of the package list table is

shown in Table 7-11.

Table 7-11 Configuration of package list table

Data strcture	Number of bit	Data notation
Package_List_Table () {		
table_id	8	uimsbf
version	8	uimsbf
length	16	uimsbf
num_of_package	8	uimsbf
for (i=0; i<N; i++) {		
MMT_package_id_length	8	uimsbf
for (j=0; j<M; j++) {		
MMT_package_id_byte	8	bslbf
}		
MMT_general_location_info ()		
}		
num_of_ip_delivery	8	uimsbf
for (i=0; i<N; i++) {		
transport_file_id	32	uimsbf
location_type	8	uimsbf
if (location_type == 0x01) {		
ipv4_src_addr	32	uimsbf
ipv4_dst_addr	32	uimsbf
dst_port	16	uimsbf
}		
if (location_type == 0x02) {		
ipv6_src_addr	128	uimsbf
ipv6_dst_addr	128	uimsbf
dst_port	16	uimsbf
}		
if (location_type == 0x05) {		
URL_length	8	uimsbf
for (j=0; j<M; j++) {		
URL_byte	8	char
}		
}		
descriptor_loop_length	16	uimsbf
for (j=0; j<M; j++) {		
descriptor ()		
}		
}		
}		

The meaning of package list table:

num_of_package (number of packages): This represents the number of packages which write the location information in this table.

MMT_package_id_length (package ID length): This represents the length of the package ID byte by the unit of byte.

MMT_package_id_byte (package ID byte): This represents the package ID.

MMT_general_location_info (location information): This represents the location information which transmits the PA message of package designated by the package ID.

num_of_ip_delivery (number of IP delivery flow): This represents the number of IP services whose location information is recorded in this table.

transport_file_id (transport file identification): This represents the label to uniquely identify file to be transmitted.

location_type (location type): This represents the kind of location information. 0x01 represents IPv4 data flow, 0x02 represents IPv6 data flow, and 0x05 represents URL.

ipv4_src_addr (source IPv4 address): This represents the source address of IPv4 data flow.

ipv4_dst_addr (destination IPv4 address): This represents the destination address of IPv4 data flow.

dst_port (destination port number): This represents the destination port number.

ipv6_src_addr (source IPv4 address): This represents the source address of IPv6 data flow.

ipv6_dst_addr (destination IPv4 address): This represents the destination address of IPv6 data flow.

URL_length (URL length): This represents the byte length of URL in case of representing location information by URL.

URL_byte (URL byte): This represents URL for IP service.

descriptor_loop_length (descriptor length): This represents the byte length of all bytes of the following descriptor.

descriptor (descriptor region): This shall be the region for the descriptor which represents detailed information of IP service.

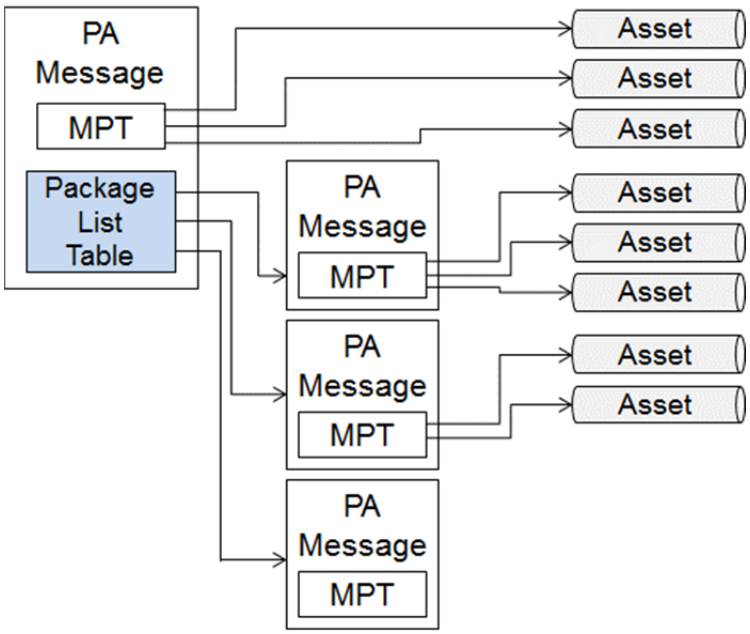


Fig. 7-2 Referring to MPT of package by package list table

MMTP packet whose packet ID is 0x0000 represents to transmit the PA message ("PA Message" shown in the left side of Fig. 7-2). In case of multiplexing plural packages, the package list table is involved in this PA massage. The package list table gives the list of packet ID for MMTP packet which transmits the PA massage involving MPT of the other packages. Therefore, by analyzing the package list table, the MMTP packet that transmits the PA massage involving MPT which is the entry point of the service can be identified by the package ID.

7.3.3.3 Layout Configuration Table (LCT)

The layout configuration table is used for connecting the layout information for presentation with the layout number. The descriptor which is stored in the layout configuration table shall be the descriptor provided in this standard. The configuration of the layout configuration table is shown in Table 7-12.

Table 7-12 Configuration of layout configuration table

Data strcture	Number of bit	Data notation
Layout_Configuration_Table () {		
table_id	8	uimsbf
version	8	uimsbf
length	16	uimsbf
number_of_loop	8	uimsbf
for (i=0; i<N; i++) {		
layout_number	8	uimsbf
device_id	8	uimsbf
number_of_region	8	uimsbf
for (j=0; j<M; j++) {		
region_number	8	uimsbf
left_top_pos_x	8	uimsbf
left_top_pos_y	8	uimsbf
right_down_pos_x	8	uimsbf
right_down_pos_y	8	uimsbf
layer_order	8	uimsbf
}		
}		
descriptor ()		
}		

The meaning of Layout Configuration Table:

number_of_loop (number of layout devices): This represents a total number of the combination of the layout and the device which are set in this table.

layout_number (layout number): This represents the layout number. '0' shall be default for the layout configuration. It shall be possible that one layout is composed of multiple devices.

device_id (device ID): This represents the number of client terminal. '0' shall be the main device.

number_of_region (number of region): This represents the number of resion in the concerned device of the concerned layout.

region_number (number of device): This represents the region number. '0' shall be default for the region number. Different region number is assigned to the different device of the same layout.

left_top_pos_x (left top horizontal position): The left top horizontal position of the region is represented as the ratio to the number of all pixels in the horizontal direction. The left side of full screen shall be 0, and the right side of full screen shall be 100.

left_top_pos_y (left top vertical position): The left top vertical position of the region is represented as the ratio to the number of all pixels in the vertical direction. Upside of full screen shall be 0, and downside of full screen shall be 100.

right_down_pos_x (right down horizontal position): The right down horizontal position of the region is represented as the ratio to the number of all pixels in the horizontal direction.

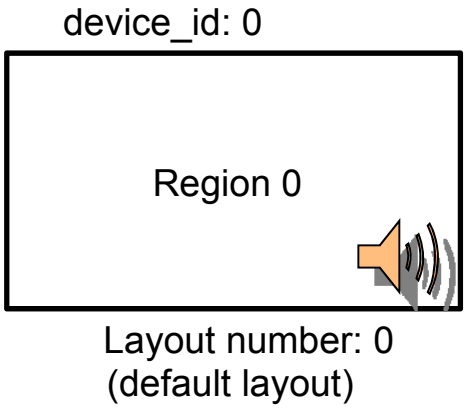
right_down_pos_y (right down vertical position): The right down vertical position of the region is represented as the ratio to the number of all pixels in the vertical direction.

layer_order (layer order): This represents the relative position of depth direction in the region. 0 represents the region represented for the most behind, and the more the number is, the more front representation is.

descriptor (descriptor region): This shall be the region for the descriptor which represents the detailed information of layout.

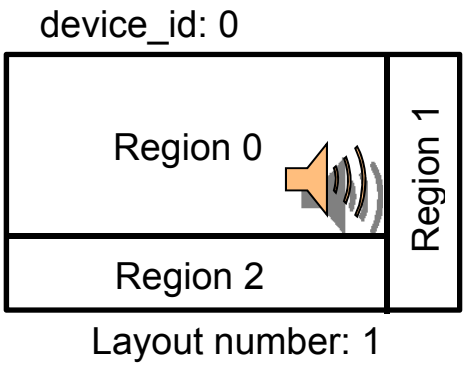
Some examples of the assignment of layout to the layout number and the values of each field by using the layout configuration table are shown in the following.

(1) a single region



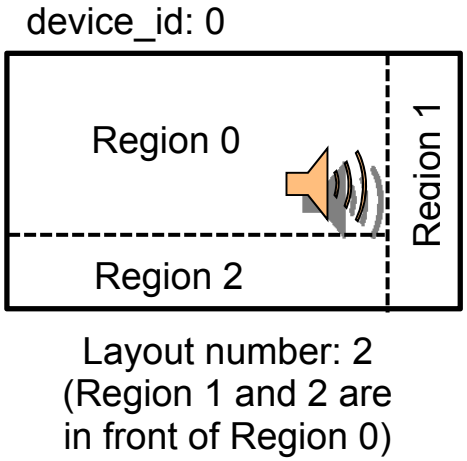
number_of_loop	1
layout_number	0
device_id	0
number_of_region	1
region_number	0
left_top_pos_x	0
left_top_pos_y	0
right_down_pos_x	100
right_down_pos_y	100
layer_order	0

(2) three regions without overlappinng



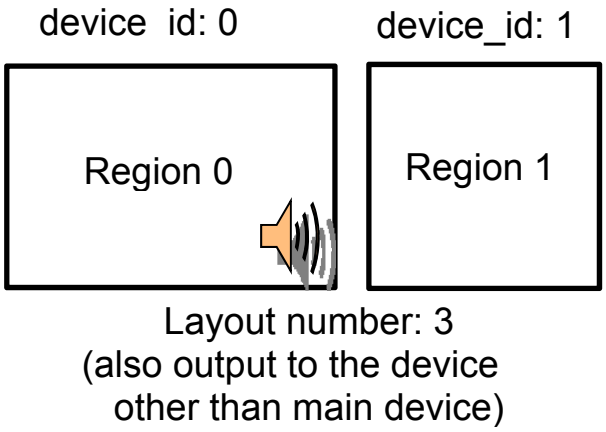
number_of_loop		1
	layout_number	1
	device_id	0
	number_of_region	3
		region_number 0
		left_top_pos_x 0
		left_top_pos_y 0
		right_down_pos_x 80
		right_down_pos_y 80
		layer_order 0
		region_number 1
		left_top_pos_x 80
		left_top_pos_y 0
		right_down_pos_x 100
		right_down_pos_y 100
		layer_order 0
		region_number 2
		left_top_pos_x 0
		left_top_pos_y 80
		right_down_pos_x 80
		right_down_pos_y 100
		layer_order 0

(3) three regions with overlappinng



number_of_loop	1
layout_number	2
device_id	0
number_of_region	3
region_number	0
left_top_pos_x	0
left_top_pos_y	0
right_down_pos_x	100
right_down_pos_y	100
layer_order	0
region_number	1
left_top_pos_x	80
left_top_pos_y	0
right_down_pos_x	100
right_down_pos_y	100
layer_order	1
region_number	2
left_top_pos_x	0
left_top_pos_y	80
right_down_pos_x	80
right_down_pos_y	100
layer_order	1

(4) two devices



number_of_loop	2
layout_number	3
device_id	0
number_of_region	1
region_number	0
left_top_pos_x	0
left_top_pos_y	0
right_down_pos_x	100
right_down_pos_y	100
layer_order	0
layout_number	3
device_id	1
number_of_region	1
region_number	1
left_top_pos_x	0
left_top_pos_y	0
right_down_pos_x	100
right_down_pos_y	100
layer_order	0

7.3.3.4 Entitlement Control Message（ECM）

[Note] ECM is also provided in Notification.

ECM is the common information which is composed of the program information and the control information, and distributes the key information to descramble, etc. The configuration of ECM is shown in Table 7-13. This item is also provided in ARIB STD-B61.

Table 7-13 Configuration of ECM

Data strcture	Number of bit	Data notation
Entitlement_Control_Message (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
ECM_data	8×N	bslbf
CRC_32	32	rpchof
}		

The meaning of ECM:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of section from immediately after section length field to the end of section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

table_id_extension (table identification extension): This shall be the region where the table identification extension is stored.

version_number (version number): This is the version number of sub-table. When information in sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0.

current_next_indicator (current next indicator): This shall be '1'.

section_number (section number): This represents the section number.

last_section_number (last section number): This specifies the last section number of sub-table which the section belongs to.

ECM_data (ECM body): This shall be the region where the ECM body is stored.

7.3.3.5 Entitlement Management Message (EMM)

[Note] EMM is also provided in Notification.

EMM transmits the individual information which includes contract information of each subscriber and key information to solve cipher of ECM (common information). Configuration of EMM is shown in Table 7-14. This item is also provided in ARIB STD-B61.

Table 7-14 Configuration of EMM

Data strcture	Number of bit	Data notation
Entitlement_Management_Message (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for (i=0; i<M; i++) {		
EMM_data	8×N	bslbf
}		
CRC_32	32	rpchof
}		

The meaning of EMM:

section_syntax_indicator (section syntax indicator): Section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of section from immediately after section length field to the end of section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

table_id_extension (table identification extension): This shall be the region where table identification extension is stored.

version_number (version number): This is the version number of sub-table. When information in sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0.

current_next_indicator (current next indicator): This shall be '1'.

section_number (section number): This represents the section number.

last_section_number (last section number): This specifies the last section number of sub-table which the section belongs to.

EMM_data (EMM body): This shall be the region where EMM body is stored.

7.3.3.6 Download Control Message (DCM)

DCM is provided in ARIB STD-B61. DCM transmits key related information which is composed of key to decode the channel cipher for download and so on. Configuration of DCM is shown in Table 7-15.

Table 7-15 Configuration of DCM

Data strcture	Number of bit	Data notation
Download_Control_Message (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
DCM_data	8×N	bslbf
CRC_32	32	rpchof
}		

The meaning of DCM:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of section from immediately after section length field to the end of section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

table_id_extension (table identification extension): This shall be the region where the table identification extension is stored.

version_number (version number): This is the version number of the sub-table. When the information in the sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0.

current_next_indicator (current next indicator): This shall be '1'.

section_number (section number): This represents the section number.

last_section_number (last section number): This specifies the last section number of the sub-table which the section belongs to.

DCM_data (DCM body): This shall be the region where DCM body is stored.

7.3.3.7 Download Management Message (DMM)

DMM is provided in ARIB STD-B61. DMM transmits key related the information which is composed of the key to decode DCM cipher and so on. The configuration of DMM is shown in Table 7-16.

Table 7-16 Configuration of DMM

Data strcture	Number of bit	Data notation
Download_Management_Message (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for (i=0; i<M; i++) {		
DMM_data	8×N	bslbf
}		
CRC_32	32	rpchof
}		

The meaning of DMM:

section_syntax_indicator (section syntax indicator): Section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of section from immediately after section length field to the end of section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

table_id_extension (table identification extension): This shall be the region where table identification extension is stored.

version_number (version number): This is the version number of sub-table. When information in sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0.

current_next_indicator (current next indicator): This shall be '1'.

section_number (section number): This represents the section number.

last_section_number (last section number): This specifies the last section number which the section belongs to.

DMM_data (DMM body): This shall be the region where DMM body is stored.

7.3.3.8 CA Table (CAT) (MH)

[Note] CA table is also provided in Notification.

The CA table is used to store the descriptor for identifying conditional access system. The descriptors which are stored in the CA table shall be the descriptors which are provided in this standard. The configuration of the CA table is shown in Table 7-17.

Table 7-17 Configuration of CA table

Data strcture	Number of bit	Data notation
Conditional_Access_Table () {		
table_id	8	uimsbf
version	8	uimsbf
length	16	uimsbf
for (i=0; i<N; i++) {		
descriptor ()		
}		
}		

The meaning of CA Table:

descriptor (descriptor region): This shall be the region for the descriptor which represents detailed the information of conditional access system.

7.3.3.9 MH-Event Information Table (MH-EIT)

MH-EIT is the time sequential information about event which is involved in each service. The configuration of EIT is shown in Table 7-18.

EIT is classified to two classes, and is discriminated by the table identification;

Self TLV stream, present/next event information,

Self TLV stream, schedule information of event.

The Event [present/next] table includes the information which is related to present event and next event in time axis which are transmitted by given the service of self TLV stream. Also, the information of the event after next can be included.

The event [schedule] table of the self TLV stream includes the event table in schedule format, namely the event after next. The transmission of the EIT [schedule] table is optional. The event information must be arranged in time sequence.

The descriptor which is stored in MH-EIT shall be the descriptor which is provided in this standard.

Table 7-18 Configuration of MH-Event Information table

Data strcture	Number of bit	Data notation
MH-Event_Information_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
service_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
tlv_stream_id	16	uimsbf
original_network_id	16	uimsbf
segment_last_section_number	8	uimsbf
last_table_id	8	uimsbf
for(i=0; i<N; i++){		
event_id	16	uimsbf
start_time	40	bslbf
duration	24	uimsbf
running_status	3	uimsbf
free_CA_mode	1	bslbf
descriptors_loop_length	12	uimsbf
for(i=0; i<N; i++) {		
descriptor ()		
}		
}		
CRC_32	32	rpchof
}		

The meaning of MH-Event information table:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of the section from immediately after the section length field to the end of the section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

service_id (service identification): This is a field of 16 bits, and acts as the label to identify this service from the other services in the TLV stream.

version_number (version number): This field of 5 bits is the version number of sub-table. When the information in sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0 next. When current next indicator is '1', the version number becomes the version number of sub-table that is defined by the table identification and the service identification, and is effective now. When current next indicator is '0', the version number becomes the version number of sub-table that is defined by the table identification and the service identification, and is effective next.

current_next_indicator (current next indicator): This indicator of one bit represents that sub-table is the present sub-table when it is '1'. When it is '0', it represents that the sub-table to be transmitted is not applied yet and used as the next sub-table.

section_number (section number): This field of 8 bits represents the number of section. The section number of first section in sub-table is 0x00. One is added to the section number for each addition of the section which has the same table identification, the same service identification, the same TLV stream identification, or the same original network identification. In this case, sub-table may be constructed as several segments. In each segment, one is added to the section number every time the section is added, and there may be the gap of the number between last section of the segment and first section of the adjacent segment.

last_section_number (last section number): This field of 8 bits specifies the last section number of sub-table which the section belongs to (namely, the section with the maximum section number).

tlv_stream_id (TLV stream identification): This is a field of 16 bits, and acts as a label to identify this TLV stream shown by EIT from the other multiplexes in the delivery system.

original_network_id (original network identification): This field of 16 bits acts as a label to specify network identification of original delivery system.

segment_last_section_number (segment last section number): This field of 8 bits specifies last section number of this segment in sub-table. For sub-table which is not divided, this field must be set to the same value as last section number (last_section_number) field.

last_table_id (last table identification): This field of 8 bits represents the last table identification being used. When only one table is used, the table identification of this table is set to this field. The information must be in the order of time scales over table identification values.

event_id (event identification): This field of 16 bits represents the identification number of event which is recorded (this is uniquely assigned in one service).

start_time (start time): This field of 40 bits represents the start time of event by Japan Standard Time (JST) and Modified Julian Day (MJD). In this field, lower 16 bits of MJD is coded by 16 bits, and the following 24 bits is coded by six binary coded decimals (BCD) of 4 bits. When start time is not defined (for example, NVOD standard service, etc.), all bits of this field are set to '1'.

Ex: 93/10/13 12:45:00 is coded as "0xC079124500".

duration (duration time): This is a field of 24 bits, and represents duration time of event by hour, minute, and second. When duration time is not defined (for example, end time is undecided because of emergency news, etc.), all bits of this field are set to '1'.

Form: six BCD codes of 4 bits = 24 bits

Ex 2: 01:45:30 is coded as "0x014530".

running_status (running status): This field of 3 bits represents the state of event which is defined in Table 7-19.

Table 7-19 State of service

value	meaning
0	undefined
1	In non-operation
2	It will start within several seconds (ex: video recording use)
3	Out of operation
4	In operation
5 – 7	Reserved for use in the future

free_CA_mode (scramble): This field of one bit represents that all component streams in the event are not scrambled when it is '0'. When it is '1', it represents that access to more than or equal to one stream is controlled by CA system.

descriptors_loop_length (descriptors loop length): This is a field of 12 bits, and specifies all byte length of the following descriptor.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.220.

7.3.3.10 MH-Common Data Table (MH-CDT)

MH-common data table is used for transmitting common data by the section format which should be stored in non-volatile memory, intending for all the receivers which receive it. The configuration of MH-common data table is shown in Table 7-20.

Table 7-20 Configuration of MH-common data table

Data strcture	Number of bit	Data notation
MH-Common_Data_Table () {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
download_data_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
original_network_id	16	uimsbf
data_type	8	uimsbf
reserved_future_use	4	bslbf
descriptors_loop_length	12	uimsbf
for (i=0; i<N; i++) {		
descriptor ()		
}		
for (j=0; j<m; j++) {		
data_module_byte	8	uimsbf
}		
CRC_32	32	rpchof
}		

The meaning of MH-Common Data Table:

section_syntax_indicator (section syntax indicator): Section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of the section from immediately after the section length field to the end of section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

download_data_id (download data identification): This field of 16 bits specifies the download data identification of common data to all the receivers. The download data identification shall be unique for each TLV stream identification. This value agrees with the value of download_data_id which is recorded in MH-logo transmission descriptor arranged in MH-SDT.

version_number (version number): This field of 5 bits is the version number of the sub-table. When the information in the sub-table is changed, one is added to the version number. When the value becomes 31, it will return to 0.

current_next_indicator (current next indicator): This indicator of one bit represents that sub-table is the present sub-table when it is '1'.

section_number (section number): This field of 8 bits represents the section number.

last_section_number (last section number): This field of 8 bits specifies the last section number in the sub-table which the section belongs to.

original_network_id (original network identification): This field of 16 bits plays a part of label to specify the network identification of the original delivery system.

data_type (attribute of data): This field of 8 bits represents the kind of download data being transmitted. 0x01 shall be the logo data, and the others are reserved for extension in the future.

descriptors_length (descriptor length): The field of 12 bits represents all byte length of the following descriptor.

data_module_byte (data module byte): The download data is recorded by syntax which is defined by each data_type.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.220.

7.3.3.11 MH-Broadcaster Information Table (MH-BIT)

MH-broadcaster information table is used for representing broadcaster information which exists on network. The descriptor which is stored in MH-BIT shall be the descriptor provided in this standard. The configuration of MH-broadcaster information table is shown in Table 7-21.

Table 7-21 Configuration of MH-broadcaster information table

Data strcture	Number of bit	Data notation
MH-Broadcaster_Information_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
original_network_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
reserved_future_use	3	bslbf
broadcast_view_propriety	1	bslbf
first_descriptors_length	12	uimsbf
for (i=0; i<N1; i++) {		
descriptor ()		
}		
for (j=0; j<N2; j++) {		
broadcaster_id	8	uimsbf
reserved_future_use	4	bslbf
broadcaster_descriptors_length	12	uimsbf
for(k=0;k<N3;k++){		
descriptor ()		
}		
}		
CRC_32	32	rpchof
}		

The meaning of MH-Broadcaster information table:

section_syntax_indicator (section syntax indicator): Section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of section from immediately after the section length field to the end of the section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

original_network_id (original network identification): This field of 16 bits acts as a label to specify the network identification of the original delivery system.

version_number (version number): This field of 5 bits is the version number of the sub-table. When information in the sub-table is changed, one is added to the version number. When the value becomes 31, it will return to 0 next.

current_next_indicator (current next indicator): This indicator of one bit represents that the sub-table is the present sub-table when it is '1'.

section_number (section number): This field of 8 bits represents the number of section.

last_section_number (last section number): This field of 8 bits specifies the last section number in the sub-table which the section belongs to.

This plays a role of label to specify the network identification of the original delivery system.

broadcast_view_propriety (whether display of broadcaster is suited or not): This indication by one bit represents that it is appropriate to present to user by the unit of broadcaster name when it is '1'. When it is '0', it represents that it is not appropriate to present to user by the unit of broadcaster name. (Each setting based on broadcaster ID in transmission is effective.)

first_descriptors_length (first descriptors length): This field of 12 bits represents all byte length of the following descriptors.

broadcaster_id (broadcaster identification): This field of 8 bits identifies broadcaster recorded in the concerned loop.

broadcaster_descriptors_length (broadcaster descriptors length): This field of 12 bits represents all byte length of the following descriptors.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.220.

7.3.3.12 MH-Software Download Trigger Table (MH-SDTT)

The software download trigger table is used for notice information of download. The configuration of the software download trigger table is shown in Table 7-22.

Table 7-22 Configuration of MH-software download trigger table

Data strcture	Number of bit	Data notation
MH-Software_Download_Trigger_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
table_id_extension	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
tlv_stream_id	16	uimsbf
original_network_id	16	uimsbf
service_id	16	uimsbf
num_of_contents	8	uimsbf
for(i=0; i<N; i++){		
group	4	uimsbf
target_version	12	uimsbf
new_version	12	uimsbf
download_level	2	bslbf
version_indicator	2	bslbf
content_description_length	12	uimsbf
reserved	4	bslbf
schedule_description_length	12	uimsbf
schedule_time-shift_information	4	uimsbf
for (j=0; j<M; j++) {		
start_time	40	uimsbf
duration	24	uimsbf
}		
for (j=0; j<M2; j++) {		
descriptors ()		
}		
}		
CRC_32	32	rpchof
}		

The meaning of MH-Software Download Trigger Table:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits. This specifies the number of bytes of the section from immediately after the section length field to the end of the section including CRC. In order that the length of all sections does not exceed 4096 bytes, the section length must not exceed 4093.

table_id_extension (table identification extension): This shall be the region to store maker_id (8 bits) and model_id (8 bits).

version_number (version number): This is the version number of the sub-table. When information in the sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0.

current_next_indicator (current next indicator): This shall be '1'.

section_number (section number): This represents the section number.

last_section_number (last section number): This specifies the last section number in the sub-table which the section belongs to.

tlv_stream_id (TLV stream identification): This is a label to identify TLV stream from the other multiplexes in the delivery system.

original_network_id (original network identification): This is a label to specify the network identification of the original delivery system.

service_id (service identification): This is a label to identify service in which download content is transmitted in download.

num_of_contents (number of download contents): This represents the number of download contents which are noticed in this table.

group (group): group_id is stored here.

target_version (target version): This represents the version number of content which shall be updated in download.

new_version (new version): This represents the version number of content which is downloaded this time in download.

download_level (download level): In case of 01, this represents forced download, and in case of 00, this represents optional download.

version_indicator (version indication): This shall be as follows:

00: Targets are all versions (version indication is invalid).

01: Target is after indicated version.

02: Target is before indicated version.

03: Target is only indicated version.

content_description_length (content description length): This represents the total byte length of schedule loop and descriptor loop.

schedule_description_length (schedule description length): This represents the byte length of schedule loop. It is common data to all the receivers, and when this value is 0, it represents that aimed download content is transmitted.

schedule_time-shift_information (schedule time-shift information): This shall be as the following.

0: By using multiple service ID's, the same download content is transmitted with the same schedule.

1-12: By using multiple service ID's, the same download content is transmitted at each service ID with time-shift of 1 to 12 hours.

13-14: Reserved.

15: Download content is transmitted by a single service_id.

start_time (start time): The start time of download delivery is shown by Japan Standard Time (JST) and Modified Julian Day (MJD).

duration (duration): duration time of the delivery

7.3.3.13 MH-Service Description Table (MH-SDT)

Each sub-table of MH-SDT represents service which is involved in a specific TLV stream. Service is either a part of self TLV stream or a part of the other TLV stream, and these are identified by the table identification. Descriptors which are stored in MH-SDT shall be the descriptors which are provided in this standard. The configuration of MH-SDT is shown in Table 7-23.

Table 7-23 Configuration of MH-service description table

Data strcture	Number of bit	Data notation
MH-Service_Description_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
tlv_stream_id	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
original_network_id	16	uimsbf
reserved_future_use	8	bslbf
for(i=0; i<N; i++){		
service_id	16	uimsbf
reserved_future_use	3	bslbf
EIT_user_defined_flags	3	bslbf
EIT_schedule_flag	1	bslbf
EIT_present_following_flag	1	bslbf
running_status	3	uimsbf
free_CA_mode	1	bslbf
descriptors_loop_length	12	uimsbf
for (j=0; j<N; j++) {		
descriptor ()		
}		
}		
CRC_32	32	rpchof
}		

The meaning of MH-Service Description Table:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and shall be always '1'.

section_length (section length): This is a field of 12 bits, and two bits of the head shall be always '00'. This specifies the number of bytes of section immediately after the section length field to the last of section involving CRC. In order that the length of all sections does not exceed 1024 bytes, the section length must not exceed 1021.

tlv_stream_id (TLV stream identification): This is a field of 16 bits, and plays a role of label to identify TLV stream represented by MH-SDT from the other multiplexes in the delivery system.

version_number (version number): This field of 5 bits is the version number of the sub-table. When information in the sub-table is changed, one is added to the version number. When the value of the version number becomes 31, it will return to 0 next. When current next indicator is

'1', the version number becomes the version number of present sub-table that is defined by the table identification and the service identification. When current next indicator is '0', the version number becomes the version number of next sub-table that is defined by the table identification and the service identification.

current_next_indicator (current next indicator): This indicator of one bit represents that the sub-table is the present sub-table when it is '1'. When it is '0', it represents that the sub-table to be transmitted is not applied yet and is used as next sub-table.

section_number (section number): This field of 8 bits represents the section number. The section number of first section in the sub-table is 0x00. One is added to the section number for each addition of the section which has the same table identification, the same service identification, or the same original network identification.

last_section_number (last section number): This field of 8 bits specifies the last section number of the sub-table that the section belongs to (namely, the section with the maximum section number).

original_network_id (original network ID): This field of 16 bits acts as a label to specify the network identification in the original delivery system.

service_id (service ID): This is a field of 16 bits, and acts as a label to identify this service from the other services in the TLV stream.

EIT_user_defined_flags (EIT broadcaster defined flag): This field of three bits can be defined independently by broadcasters as an extension representing whether EIT is transmitted or not. When it is not used, it shall be '111'.

EIT_schedule_flag (EIT [schedule] flag): This field of one bit represents that EIT [schedule] information of the service exists in present TLV stream when it is '1'. When this flag is '0', it represents that EIT [schedule] information of the service does not exist in present TLV stream.

EIT_present_following_flag (EIT [present/next] flag): This field of one bit represents that EIT [present/next] information of the service exists in present TLV stream when it is '1'. When this flag is '0', it represents that EIT [present/next] information of the service does not exist in present TLV stream.

running_status (running status): This field of three bits represents the running status which is defined in Table 7-24.

Table 7-24 State of service

value	meaning
0	undefined
1	In non-operation
2	It will start within several seconds (ex: video recording use)
3	Out of operation
4	In operation
5 – 7	Reserved for use in the future

free_CA_mode (scramble): This field of one bit represents that all component streams in the event are not scrambled when it is '0'. When it is '1', it represents that access to more than or equal to one stream is controlled by CA system.

descriptors_loop_length (descriptors loop length): This is a field of 12 bits, and specifies all byte length of the following descriptors.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.222.0.

7.3.3.14 MH-Time Offset Table (MH-TOT)

MH-TOT transmits the information of JST time and date (Modified Julian Day). Descriptors which are stored in MH-TOT shall be the descriptors that are provided in this standard. The configuration of MH-TOT is shown in Table 7-25.

Table 7-25 Configuration of MH-time offset table

Data strcture	Number of bit	Data notation
MH-Time_Offset_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
JST_time	40	bslbf
reserved	4	bslbf
descriptors_loop_length	12	uimsbf
for (i=0; i<N; i++){		
descriptor ()		
}		
CRC_32	32	rpchof
}		

The meaning of MH-Time offset table:

section_syntax_indicator (section syntax indicator): This is an indicator of one bit, and is set to '0'.

section_length (section length): This is a field of 12 bits, and two bits of the head shall be always '00'. This specifies the number of bytes of the section from immediately after section

length field to the last of section involving CRC.

JST_time (present date, present time): This field of 40 bits involves present date and present time by Japan Standard Time (JST) and Modified Julian Day (MJD). (Refer to ARIB STD-B10 Appendix C) In this field, lower 16 bits of MJD are coded by 16 bits, and the following 24 bits are coded by six binary coded decimal (BCD) of 4 bits.

descriptors_loop_length (descriptor loop length): This is a field of 12 bits, and specifies all byte length of the following descriptors.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.222.0.

7.4 Descriptor

7.4.1 Summary of Descriptor

The descriptor is the control information for providing more detailed information, and it is arranged in table.

7.4.2 Arrangement of Descriptor in Table

The arrangement of descriptor in table which is specified in this standard is shown in Table 7-26.

Table 7-26　Descripters-arranging table

Descriptor	Table							
	MPT	CAT (MH)	MH-EIT	MH-BIT	MH-SDTT	MH-SDT	LCT	MH-TOT
Asset group descriptor	○							
Event package descriptor			○					
Background color descriptor							○	
MPU presentation region descriptor	○							
MPU timestamp descriptor*1	○							
Dependency descriptor*1	○							
Access control descriptor*1	○	○						
Scramble descriptor*1	○	○						
Message authentication method descriptor	○	○						
Emergency information descriptor*1 (MH)	○							
MH-MPEG-4 Audio descriptor	○							
MH-MPEG-4 Audio extension descriptor	○							
MH-HEVC descriptor	○							
MH-Linkage descriptor	○		○			○		
MH-Event group descriptor			○					
MH-Service list descriptor				○				
MH-Short event descriptor			○					
MH-Extended event descriptor			○					
Video component descriptor	○		○					
MH-Stream identifier descriptor	○							
MH-Content descriptor			○					
MH-Parental rating descriptor	○		○					
MH-Audio component descriptor	○		○					
MH-Target region descriptor	○							
MH-Series descriptor			○					
MH-SI parameter descriptor				○				
MH-Broadcaster name descriptor				○				
MH-Service descriptor						○		

Descriptor	Table							
	MPT	CAT (MH)	MH-EIT	MH-BIT	MH-SDTT	MH-SDT	LCT	MH-TOT
IP data flow descriptor						○		
MH-CA startup descriptor	○	○						
MH-Data component descriptor	○							
MH-Local time offset descriptor								○
MH-Component group descriptor			○					
MH-Logo transmission descriptor						○		
MPU extention timestamp descriptor	○							
MPU download content descriptor					○			
MH-Network download content descriptor					○			
MH-Download protection descriptor	○				○			
Application service descriptor	○							
MH-Hierarchy descriptor	○							
Content copy control descriptor	○		○			○		
Content usage control descriptor	○		○			○		
Emergency news descriptor	○							
MH-CA contract information descriptor			○			○		
MH-CA service descriptor		○						
Related broadcaster descriptor				○				
Multimedia service information descriptor			○					

*1: Descriptor that is provided in Notification

7.4.3 Definition of Descriptor

The data structure shown in the following applies to all descriptors which are defined in this section.

descriptor_tag (descriptor tag): The descriptor tag is a field of 16 bits, and identifies each descriptor. The value of descriptor tag is defined in Table 4-10.

descriptor_length (descriptor length): This shall be the region where the number of data byte which follows to this field is written. The bit length of the descriptor length field is varied with the descriptor.

7.4.3.1 Asset Group Descriptor

The asset group descriptor supplies the group relationship of asset and priority in the group.

The configuration of the asset group descriptor is shown in Table 7-27. The asset group descriptor is arranged in the region of the asset descriptor in MP table.

Table 7-27 Configuration of asset group descriptor

Data strcture	Number of bit	Data notation
Asset_Group_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
group_identification	8	uimsbf
selection_level	8	uimsbf
}		

The meaning of Asset group descriptor:

group_identification (group ID): This represents ID which groups asset, for example, video and audio, etc.

selection_level (selection level): This represents the selection level in group. Asset whose value of selection level is '0' is selected as a default. When a default asset cannot be selected, it represents that the smaller asset in group has priority to be selected in order of the number.

7.4.3.2 Event Package Descriptor

The event package descriptor supplies correspondence between events representing program and packages. The configuration of the event package descriptor is shown in Table 7-28. The event package descriptor is arranged in MH-EIT which is transmitted by M2 section message.

Table 7-28 Configuration of event package descriptor

Data strcture	Number of bit	Data notation
Event_Package_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
MMT_package_id_length	8	uimsbf
for (i=0; i<N; i++) {		
MMT_package_id_byte	8	uimsbf
}		
}		

The meaning of event package descriptor:

MMT_package_id_length (package ID length): This represents byte length of the following MMT package ID byte region.

MMT_package_id_byte (package ID byte): This denotes MMT package ID which corresponds to the concerned event.

7.4.3.3 Background Color Descriptor

The background color descriptor supplies background color of the most rear plane in layout designation. The configuration of the background color descriptor is shown in Table 7-29.

Table 7-29 Background color descriptor

Data strcture	Number of bit	Data notation
Background_Color_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
background_color	24	uimsbf
}		

The meaning of background color descriptor:

background_color (background color): This field of 24 bits represents background color by color designation with RGB of 8 bits each.

7.4.3.4 MPU Presentation Region Descriptor

The MPU presentation region descriptor supplies the position to present MPU. The MPU presentation region descriptor shall be the configuration shown in Table 7-30. The MPU presentation region descriptor shall be arranged in the region of asset descriptor of MPT.

Table 7-30 Configuration of MPU presentation region descriptor

Data strcture	Number of bit	Data notation
MPU_Presentation_Region_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
mpu_sequence_number	32	uimsbf
layout_number	8	uimsbf
region_number	8	uimsbf
length_of_reserved	8	uimsbf
for (j=0; j<M; j++) {		
reserved_future_use	8	bslbf
}		
}		
}		

The meaning of MPU presentation region descriptor:

mpu_sequence_number (MPU sequence number): This represents the sequence number of MPU which specifies presentation region.

layout_number (layout number): This represents the layout number which presents MPU. Layout number 0 shall be the default layout.

region_number (region number): This represents the region number in layout which presents MPU. Region number 0 shall be the default region number.

length_of_reserved (length of reserved region): This represents the following field length for reservation in the future by the unit of byte.

reserved_future_use (reserved for the future use): This shall be the region for reservation in the future.

7.4.3.5 MPU Timestamp Descriptor

[Note] MPU timestamp descriptor is also provided in Notification.

The MPU timestamp descriptor represents the presentation time of first access unit by the order of presentation in MPU. The configuration of MPU timestamp descriptor is shown in Table 7-31. The MPU timestamp descriptor is arranged in the asset descriptor region of MPT.

Table 7-31 Configuration of MPU timestamp descriptor

Data strcture	Number of bit	Data notation
MPU_Timestamp_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
mpu_sequence_number	32	uimsbf
mpu_presentation_time	64	uimsbf
}		
}		

The meaning of MPU timestamp descriptor:

mpu_sequence_number (MPU sequence number): This represents the sequence number of MPU denoting timestamp.

mpu_presentation_time (MPU presentation time): This represents the presentation time of MPU by the NTP timestamp format of 64 bits.

7.4.3.6 Dependency Descriptor

[Note] Dependency descriptor is also provided in Notification.

The dependency descriptor supplies asset ID of asset which has dependency relationship. The configuration of the dependency descriptor is shown in Table 7-32. The dependency descriptor is arranged in the region for asset descriptor of MPT.

Table 7-32 Configuration of dependency descriptor

Data strcture	Number of bit	Data notation
Dependency_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	16	uimsbf
num_dependencies	8	uimsbf

for (i=0; i<N; i++) {		
asset_id_scheme	32	uimsbf
asset_id_length	8	uimsbf
for (j=0; j<M; j++) {		
asset_id_byte	8	uimsbf
}		
}		
}		

The meaning of dependency descriptor:

num_dependencies (number of dependency assets): This represents the number of assets which are complementary to the asset which this descriptor is inserted in.

asset_id_scheme (asset ID format): This represents the format of asset ID of complementary asset.

asset_id_length (asset ID length): This represents the byte length of the asset ID of complementary asset by the unit of byte.

asset_id_byte (asset ID byte): This represents asset ID of complementary asset.

7.4.3.7 Access Control Descriptor

[Note] Access control descriptor is also provided in Notification.

The access control descriptor supplies information to identify conditional access system. The configuration of access control descriptor is shown in Table 7-33.

Table 7-33 Configuration of access control descriptor

Data strcture	Number of bit	Data notation
Access_Control_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
CA_system_ID	16	uimsbf
MMT_general_location_info ()		
private_data	8xN	bslbf
}		

The meaning of access control descriptor:

CA_system_ID (conditional access system identifier): This represents the kind of conditional access system.

MMT_general_location_info (): This represents the location of MMTP packet which involves related information. When this is arranged in CAT (MH), it represents the location of EMM, and when it is arranged in MPT, it represents the location of ECM.

private_data (private data): This shall be the region where data is written.

7.4.3.8 Scrambler Descriptor

[Note] Scrambler descriptor is also provided in Notification.

The scrambler descriptor supplies information to identify the object for encryption and the kind of algorithm in scramble. The configuration of the scrambler descriptor is shown in Table 7-34.

Table 7-34 Configuration of scrambler descriptor

Data strcture	Number of bit	Data notation
Scrambler_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
layer_type	2	uimsbf
reserved	6	bslbf
scramble_system_id	8	uimsbf
private_data	8xN	bslbf
}		

The meaning of scramble system descriptor:

layer_type (target layer identifier): This represents the encryption object for scramble. '01' represents MMTP packet is the object, and '10' represents IP packet is the object.

scramble_system_id (scramble system identifier): This represents the kind of encryption algorithm for scramble. '00000001' represents AES with a key length of 128 bits, and '00000010' represents Camellia with a key length of 128 bits.

private_data (private data): This shall be the region where data is written.

7.4.3.9 Message Authentication Method Descriptor

The message authentication method descriptor supplies information to identify the message authentication system when message is authenticated. Also, when message is not authenticated, this descriptor is not arranged. The configuration of the message authentication method descriptor is shown in Table 7-35.

Table 7-35 Configuration of message authentication method descriptor

Data strcture	Number of bit	Data notation
Message_Authentication_Method_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
layer_type	2	uimsbf
reserved	6	bslbf
message_authentication_system_id	8	uimsbf
private_data	8xN	bslbf
}		

The meaning of message authentication method descriptor:

layer_type (object layer identifier): This represents the encryption object for scramble. '01'

represents MMTP packet is the object, and '10' represents IP packet is the object.

message_authentication_system_id (message authentication system identifier): This represents the kind of message authentication system which detects falsification of MMTP packet or IP packet.

private_data (private data): This shall be the region where data is written.

7.4.3.10 Emergency Information Descriptor (MH)

[Note] Emergency information descriptor is also provided in Notification.

The emergency information descriptor is a signal based on the emergency alarm signal provided in Radio Equipment Rule, Article 9, Item 3, No.5, and is used when emergency alarm broadcasting is operated. The configuration of the emergency information descriptor is shown in Table 7-36.

Table 7-36 Configuration of emergency information descriptor

Data strcture	Number of bit	Data notation
Emergency_Information_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
service_id	16	uimsbf
start_end_flag	1	bslbf
signal_level	1	bslbf
reserved_future_use	6	bslbf
area_code_length	8	uimsbf
for (j=0; j<N; j++) {		
area_code	12	bslbf
reserved_future_use	4	bslbf
}		
}		
}		

The meaning of emergency information descriptor:

service_id (service identification): This field of 16 bits represents the broadcasting program number.

start_end_flag (start/end flag): This flag of one bit corresponds to start flag and end flag, among emergency alarm signals which are provided in Notification of the Ministry of Post and Telecommunications No. 405, 1985. When this bit is '1', it represents that the emergency alarm signal is started or on the air. When this bit is '0', it represents that the emergency alarm signal is ended.

signal_level (kind of signal): This field of one bit corresponds to the kind of the emergency alarm signal which is provided in Operational Rule of Radio Station, Article 138-2. When this bit is '0', it represents the emergency alarm signal which is transmitted is class 1 start signal. When this bit is '1', it represents the emergency alarm signal which is transmitted is class 2 start signal.

area_code_length (area code length): This is a field of 8 bits, and represents the byte length of the following area code.

area_code (area code): This is a field of 12 bits, and corresponds to the area code which is provided in Operational Rule of Radio Station, Article 138-3. For assignment of area code, those which is provided in Notification of the Ministry of Post and Telecommunications No. 405, 1985 is used.

7.4.3.11 MH-MPEG-4 Audio Descriptor

The MH-MPEG-4 Audio descriptor is used for describing basic information to specify the coding parameters of audio stream by ISO/IEC 14496-3 (MPEG-4 audio). The configuration of MH-MPEG-4 audio descriptor is shown in Table 7-37.

Table 7-37 Configuration of MH-MPEG-4 audio descriptor

Data strcture	Number of bit	Data notation
MH-MPEG-4_Audio_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
MPEG-4_audio_profile_and_level	8	uimsbf
}		

The meaning of MH-MPEG-4 audio descriptor:

MPEG-4_audio_profile_and_level (profile and level of MPEG-4 audio): This is a field of 8 bits, and represents the profile and level of MPEG-4 audio stream. As for the assignment of value in this field, refer to Rec. ITU-T H.222.0|ISO/IEC 13818-1 (2006)/Amd.1 (01/2007), Table 2-71. When this value is 0x0F, it represents the profile and level are not provided. When this value is 0xFF, it represents the profile and level are those which are not provided in this field, and the profile and level are described by the MH-MPEG-4 audio extension descriptor which is provided in 7.4.3.12.

7.4.3.12 MH-MPEG-4 Audio Extension Descriptor

The MH-MPEG-4 audio extension descriptor is used for describing the profile and level of MPEG-4 audio stream, and specific setting for the coding system. The configuration of MH-MPEG-4 audio extension descriptor is shown in Table 7-38.

Table 7-38 Configuration of MH-MPEG-4 audio extension descriptor

Data strcture	Number of bit	Data notation
MH-MPEG-4_Audio_Extension_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
ASC_flag	1	bslbf
reserved	3	bslbf
num_of_loops	4	uimsbf
for(i=0 ;i< N ; i++){		
audioProfileLevelIndication	8	uimsbf
}		
if (ASC_flag == 1){		
ASC_size	8	uimsbf
AudioSpecificConfig ()		
}		
}		

The meaning of MH-MPEG-4 audio extension descriptor:

ASC_flag (ASC flag): When ASC_size is involved in this descriptor, this field shall be '1'.

num_of_loops (number of loops): This is a field of four bits, and represents the number of audioProfileLevelIndication which is involved immediately after this.

audioProfileLevelIndication (indication of audio profile, level): This represents the profile and level of MPEG-4 audio stream. Refer to ISO/IEC 14496-3, section 1.5.2.4. One MPEG-4 audio profile may be based on plural profiles and levels, and this descriptor can describe 15 audio profiles and levels in maximum.

ASC_size (ASC size): This is a field of 8 bits, and represents the number of byte of AudioSpecificConfig () immediately after this.

AudioSpecificConfig () (audio specific setting): This represents the specific setting for MPEG-4 audio stream. Refer to ISO/IEC 14496-3, section 1.6.2.1.

7.4.3.13 MH-HEVC Descriptor

The MH-HEVC descriptor is used for describing the basic coding parameters of video stream (HEVC stream) in ITU-T Recommendation H.265 | ISO/IEC 23008-2. The configuration of the MH-HEVC descriptor is shown in Table 7-39.

Table 7-39 Configuration of MH-HEVC descriptor

Data strcture	Number of bit	Data notation
MH-HEVC_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
profile_space	2	uimsbf
tier_flag	1	bslbf
profile_idc	5	uimsbf
profile_compatibility_indication	32	bslbf
progressive_source_flag	1	bslbf
interlaced_source_flag	1	bslbf
non_packed_constraint_flag	1	bslbf
frame_only_constraint_flag	1	bslbf
copied_44bits	44	bslbf
level_idc	8	uimsbf
temporal_layer_subset_flag	1	bslbf
HEVC_still_present_flag	1	bslbf
HEVC_24hr_picture_present_flag	1	bslbf
sub_pic_hrd_params_not_present_flag	1	bslbf
reserved	2	bslbf
HDR_WCG_idc	2	bslbf
if (temporal_layer_subset_flag == 1) {		
temporal_id_min	3	uimsbf
reserved	5	bslbf
temporal_id_max	3	uimsbf
reserved	5	bslbf
}		
}		

The meaning of MH-HEVC descriptor:

profile_space (profile space): This follows the provision of general_profile_space or sub_layer_profile_space in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

tier_flag (tier flag): This follows the provision of general_tier_flag or sub_layer_tier_flag in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

profile_idc (HEVC profile): This follows the provision of general_profile_idc or sub_layer_profile_idc in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

profile_compatibility_indication (profile compatibility): This follows the provision of general_profile_compatibility_flag[i] or sub_layer_profile_ compatibility_flag[i] in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

progressive_source_flag (progressive scanning source flag): This follows the provision of general_progressive_source_flag or sub_layer_ progressive_source_flag in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

interlaced_source_flag (interlaced scanning source flag): This follows the provision of general_interlaced_source_fla or sub_layer_ interlaced_source_flag in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

non_packed_constraint_flag (non-pack constraint flag): This follows the provision of general_non_packed_constraint_flag or sub_layer_non_packed_constraint_flag in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

frame_only_constraint_flag (frame constraint flag): This follows the provision of general_frame_only_constraint_flag or sub_layer_frame_only_constraint_flag in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

copied_44bits (copied 44 bits): The value of 44 bits between general_frame_only_constraint_ flag and general_level_idc of profile_tier_level( ) in ITU-T Recommendation H.265 | ISO/IEC 23008-2 is set.

level_idc (HEVC level): This follows the provision of general_level_idc or sub_layer_ level_idc in ITU-T Recommendation H.265 | ISO/IEC 23008-2.

temporal_layer_subset_flag (temporal layer subset flag): This represents that when this field is '1', the information of temporal layer subset is involved in this descriptor. This field must be '1' for the subset and sub-bitstream of temporal layer coding.

HEVC_still_present_flag (HEVC with still picture): When this field is '1', still picture is involved in HEVC stream. When this field is '0', HEVC stream must not involve still picture.

HEVC_24hr_picture_present_flag (HEVC 24 hours picture flag): When this field is '1', the HEVC 24 hours picture is involved in HEVC. The HEVC 24 hours picture is HEVC access unit which has a presentation time with more than 24 hours. When this field is '0', HEVC picture stream must not involve the HEVC 24 hours picture.

sub_pic_hrd_params_not_present_flag (flag without sub-picture HRD parameter): When this field is '0', sub_pic_hrd_params_not_present_flag must be '1' for VUI of HEVC stream, and when this field is '1', HEVC stream does not involve sub_pic_hrd_params_not_present_flag in VUI or it is '0'.

HDR_WCG_idc (HDR/WCG): This represents whether the video is made by high dynamic range (HDR) and wide color gamut (WCG) or not, according to Table 7-39-1.

Table 7-39-1 HDR_WDC_idc

HDR_WCG_idc	description
0	SDR (standard EOTF in ITU-R Rec. BT.1886 and color gamut in ITU-R Rec. BT.709)
1	WCG (color gamut in ITU-R Rec. BT.2020)
2	HDR and WCG
3	Without information

temporal_id_min (temporal ID minimum value): This represents the minimum value of temporal ID which is provided in ITU-T Recommendation H.265 | ISO/IEC 23008-2 in all access units which are involved in the concerned elementary stream.

temporal_id_max (temporal ID maximum value): This represents the maximum value of temporal ID which is provided in ITU-T Recommendation H.265 | ISO/IEC 23008-2 in all access units which are involved in the concerned elementary stream.

7.4.3.14 MH-Linkage Descriptor

The MH-linkage descriptor identifies service which is supplied when viewer demands additional information related to a specific matter which is written in the service information

system. It represents those which additional information can be used by the position of linkage descriptor in data structure. The CA alternative service is also identified by using linkage descriptor. In case that CA refuses to access specific matter written in SI system, this alternative service is automatically selected by the receiver. The configuration of MH-linkage descriptor is shown in Table 7-40.

Table 7-40 Configuration of MH-linkage descriptor

Data strcture	Number of bit	Data notation
MH-Linkage_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	16	uimsbf
tlv_stream_id	16	uimsbf
original_network_id	16	uimsbf
service_id	16	bslbf
linkage_type	8	uimsbf
for (i=0; i<N; i++){		
private_data_byte	8	bslbf
}		
}		

The meaning of MH-Linkage descriptor:

tlv_stream_id (TLV stream identification): This is a field of 16 bits, and identifies TLV stream in which indicated information service is involved.

original_network_id (original network identification): This field of 16 bits acts as a label which specifies the network identification of the original delivery system in the information service which is indicated.

service_id (service identification): This is a field of 16 bits, and uniquely identifies the information service in the TLV stream. When the value of linkage type is 0x04, the field of the service identification does not have a sense, and it is set to 0x0000.

linkage_type (linkage type): This is a field of 8 bits, and represents information such as linkage type. (Refer to Table 7-41.)

Table 7-41 Code of linkage type

Linkage type	Description
0x00	Reserved for the use in the future
0x01	Information service
0x02	Electronic program guide (EPG) service
0x03	CA alternative service
0x04	TLV stream including all network/bouquet SI
0x05	Alternative service
0x06	Data broadcasting service
0x07 – 0x0A	Reserved for the use in the future
0x0B	INT
0x0C – 0x7F	Reserved for the use in the future
0x80 – 0xBF	Defined by broadcasters
0xC0 – 0xFD	Reserved for the use in the future (region defined by standardization organization)
0xFE	Reserved for re-transmission
0xFF	Reserved for the use in the future

private_data_byte (private data): This is a field of 8 bits, and has the value which is defined individually.

7.4.3.15 MH-Event Group Descriptor

When there is a relationship between plural events, the MH-event group descriptor is used for representing that these event groups are grouped. The configuration of the MH-event group descriptor is shown in Table 7-42.

Table 7-42 Configuration of MH-event group descriptor

Data strcture	Number of bit	Data notation
MH-Event_Group_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
group_type	4	uimsbf
event_count	4	uimsbf
for (i=0; i< event_count; i++){		
service_id	16	uimsbf
event_id	16	uimsbf
}		
if (group_type == 4 || group_type == 5) {		
for(i=0; i<N; i++) {		
original_network_id	16	uimsbf
tlv_stream_id	16	uimsbf
service_id	16	uimsbf
event_id	16	uimsbf
}		
}		
else {		
for (i=0; i< N; i++) {		
private_data_byte	8	uimsbf
}		
}		
}		

The meaning of MH-Event group descriptor:

group_type (group type): This is a field of 4 bits, and represents the group type of event according to Table 7-43.

Table 7-43 Group type

Group type	Description
0x0	undefined
0x1	Event in common
0x2	Event relay
0x3	Event transfer
0x4	Event relay to the other network
0x5	Event transfer from the other network
0x6 – 0xF	undefined

event_count (number of event loop): This is a field of 4 bits, and represents the number of the following event_id loops.

service_id (service identification): This is a field of 16 bits, and represents a service

identification of information service to be related.

event_id (event identification): This is a field of 16 bits, and represents the event identification of event to be related.

original_network_id (original network identification): This is a field of 16 bits, and represents an identification value of network where related event is transmitted in case of event relay extending over networks, or event transfer.

tlv_stream_id (TLV stream identification): This is a field of 16 bits, and represents an identification value of TLV stream where related event is transmitted in case of event relay extending over networks, or event transfer.

7.4.3.16 MH-Service List Descriptor

The MH-service list descriptor supplies a list of service by the service identification and the service type. The configuration of the MH-service list descriptor is shown in Table 7-44.

Table 7-44 MH-service list descriptor

Data strcture	Number of bit	Data notation
MH-Service_List_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++){		
service_id	16	uimsbf
service_type	8	uimsbf
}		
}		

The meaning of MH-service list descriptor:

service_id (service identification): This is a field of 16 bits, and uniquely identifies the information service in the TLV stream.

service_type (service type): This is a field of 8 bits, and represents the kind of service according to Table 5-5.

7.4.3.17 MH-Short Event Descriptor

The MH-short event descriptor represents event name and short description of the event by the text format. The MH-short format event descriptor is shown in Table 7-45.

Table 7-45 MH-short event descriptor

Data strcture	Number of bit	Data notation
MH-Short_Event_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	16	uimsbf
ISO_639_language_code	24	bslbf
event_name_length	8	uimsbf
for (i=0; i<event_name_length; i++){		
event_name_char	8	uimsbf
}		
text_length	16	uimsbf
for (i=0; i<text_length; i++){		
text_char	8	uimsbf
}		
}		

The meaning of MH-short event descriptor:

ISO_639_language_code (language code): This field of 24 bits represents the language of the following information field in alphabetical three-letters code provided in ISO639-2. Each letter is coded in 8 bits according to ISO8859-1, and inserted to 24 bits field by the order.

Example: Japanese language is "jpn" by alphabetical three-letters code, and is coded as follows:

"0110 1010 0111 0000 0110 1110"

event_name_length (program name length): This field of 8 bits represents the byte length of the following program name.

event_name_char (program name): This is a field of 8 bits. A series of character information field represents the program name.

text_length (program description length): This field of 16 bits represents the byte length of the following program description.

text_char (program description): This is a field of 8 bits. A series of character information field describes the explanation of program.

7.4.3.18 MH-Extended Event Descriptor

The MH-extended event descriptor is used by adding to the MH-short event descriptor, and supplies detailed description of event. In order that transmission of information with more than 255 bytes length is possible on one event, more than or eaqual to one MH-extended event descriptor can be used. Document information is composed of two columns, namely, item name field and item description field. A typical application for this structure is supplying the list of cast. For example, "producer" is recorded in item name field, and the name of producer is recorded in the item description field. The configuration of the MH-extended event descriptor is shown in Table 7-46.

Table 7-46 Configuration of MH-extended event descriptor

Data strcture	Number of bit	Data notation
MH-Extended_Event_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	16	uimsbf
descriptor_number	4	uimsbf
last_descriptor_number	4	uimsbf
ISO_639_language_code	24	bslbf
length_of_items	16	uimsbf
for (i=0; i<N; i++) {		
item_description_length	8	uimsbf
for (j=0; j<item_description_length; j++){		
item_description_char	8	uimsbf
}		
item_length	16	uimsbf
for (j=0; j<item_length; j++){		
item_char	8	uimsbf
}		
}		
text_length	16	uimsbf
for (i=0; i<text_length; i++){		
text_char	8	uimsbf
}		
}		

The meaning of MH-extended event descriptor:

descriptor_number (descriptor number): This is a field of 4 bits which represents the descriptor number. This is used for connecting information which is not contained by one descriptor. The first descriptor number of the connected extended event descriptor set shall be 0x0. Each time the extended event descriptor is added in this section, the descriptor number is incremented by 1.

last_descriptor_number (last descriptor number): This field of 4 bits represents the number of the last extended event descriptor (descriptor which has the maximum descriptor number) in the connected descriptor set which this descriptor is a part of the field.

ISO_639_language_code (language code): This field of 24 bits identifies the language of component (sound or data) and the language describing letters which are involved in this descriptor. The language code is denoted by alphabetical three-letters code which is provided in ISO 639-2. Each letter is coded by 8 bits according to ISO 8859-1, and is inserted to 24 bits field by the order.

Example: Japanese language is “jpn” for alphabetical three-letters code, and is coded as the following.

“0110 1010 0111 0000 0110 1110”

length_of_item (length of items): This is a field of 16 bits, and represents the byte length of the following item.

item_description_length (length of item name): This is a field of 8 bits, and represents the byte

length of item name.

item_description_char (item name (character code)): This is a field of 8 bits, and the field of a series of item name specifies the letter description of item name.

item_length (item length): This is a field of 16 bits, and represents the byte length of item description.

item_char (item description (character code)): This is a field of 8 bits, and the field of a series of item description specifies the character description of item.

text_length (length of extended description): This is a field of 16 bits, and represents the byte length of extended description without item.

text_char (extended description (character code)): This is a field of 8 bits, and the field of a series of extended description specifies the character description without item.

7.4.3.19 Video Component Descriptor

The video component descriptor represents parameter and explanation about video component, and is used to express elementary stream by letter format. The configuration of video component descriptor is shown in Table 7-47.

Table 7-47 Configuration of video component descriptor

Data strcture	Number of bit	Data notation
Video_Component_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
video_resolution	4	uimsbf
video_aspect_ratio	4	uimsbf
video_scan_flag	1	bslbf
reserved	2	bslbf
video_frame_rate	5	uimsbf
component_tag	16	uimsbf
video_transfer_characteristics	4	uimsbf
reserved	4	bslbf
ISO_639_language_code	24	bslbf
for (i=0; i<N; i++) {		
text_char	8	uimsbf
}		
}		

The meaning of video component descriptor:

video_resolution (video signal resolution): This field of 4 bits represents the resolution of video signal in the vertical direction, and is coded according to Table 7-48.

Table 7-48　Video signal resolution

Value of the video signal resolution	meaning
0	Video signal resolution is not specified.
1	180
2	240
3	480（525）
4	720（750）
5	1080（1125）
6	2160
7	4320
8 – 15	Reserved for use in the future

video_aspect_ratio (video signal aspect ratio): This field of 4 bits represents the aspect ratio of video signal, and is coded according to Table 7-49.

Table 7-49　Video signal aspect ratio

Value of the video signal aspect ratio	meaning
0	Video signal aspect ratio is not specified.
1	4:3
2	16:9　with pan vector
3	16:9　without pan vector
4	> 16:9
5 – 15	Reserved for use in the future

video_scan_flag (video scan flag): This shall be '0' in case that video signal is interlaced, and shall be '1' in case that video signal is progressive.

video_frame_rate (video signal frame rate): This field of 5 bits represents the frame rate of video signal, and is coded in according to Table 7-50.

Table 7-50 Video signal frame rate

Value of the video signal frame rate	meaning
0	Frame rate is not specified.
1	15
2	24/1.001
3	24
4	25
5	30/1.001
6	30
7	50
8	60/1.001
9	60
10	100
11	120/1.001
12	120
13 – 31	Reserved for use in the future

component_tag (component tag): This is a field of 16 bits. The component tag is a label to identify component stream, and the value is the same as that of the component tag in the MH-stream identification descriptor (refer to the section 7.4.3.20). (Although, in case that the MH-stream identification descriptor exists in MPT.)

video_transfer_characteristics (video signal transfer characteristics): This field of 4 bits identifies the transfer characteristics of video signal, and is coded according to Table 7-51.

Table 7-51 Video signal transfer characteristics

Value of the video signal transfer characterisitics	meaning
0	Video signal transfer characterisitics is not specified.
1	transfer_characteristics of VUI = 1 (Rec. ITU-R BT.709-5)
2	transfer_characteristics of VUI = 11 (IEC 61966-2-4)
3	transfer_characteristics of VUI = 14 (Rec. ITU-R BT.2020)
4	transfer_characteristics of VUI = 16 (Rec. ITU-R BT.2100 PQ)
5	transfer_characteristics of VUI = 18 (Rec ITU-R BT.2100 HLG)
6 – 15	Reserved for use in the future

ISO_639_language_code (language code): This field of 24 bits identifies the language of component (sound, or data) and the language describing letters which are involved in this descriptor. The language code is denoted by alphabetical three-letters code which is provided in ISO 639-2. Each letter is coded by 8 bits according to ISO 8859-1, and is inserted to 24 bits field by the order.

Example: Japanese language is “jpn” for alphabetical three-letters code, and is coded as the following.

“0110 1010 0111 0000 0110 1110”

text_char (component description): This is a field of 8 bits. Fields in a series of the component description specify the character description of the component stream.

7.4.3.20 MH-Stream Identifier Descriptor

The MH-stream identifier descriptor which is used in MPT is used in order to be able to refer to the description content (for example, it is that the component stream of a service is “the number of pixels in the vertical direction is 4320, the aspect ratio is 16:9, the pan-vector is used, and the frame rate is 60/1.001.”) that is represented by the video component descriptor in MH-EIT by the label which is labeled to the component stream of service. The configuration of the MH-stream identifier descriptor is shown in Table 7-52.

Table 7-52 MH-stream identifier descriptor

Data strcture	Number of bit	Data notation
MH-Stream_Identifier_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
component_tag	16	uimsbf
}		

The meaning of MH-stream identification descriptor:

component_tag (component tag): The component stream of service can refer to the description content represented by the video component descriptor by this field of 16 bits. The value of component tag which is given to each stream should be different.

7.4.3.21 MH-Content Descriptor

The MH-Content descriptor represents the genre of the event. The configuration of the MH-content descriptor is shown in Table 7-53.

Table 7-53 MH-content descriptor

Data strcture	Number of bit	Data notation
MH-Content_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
content_nibble_level_1	4	uimsbf
content_nibble_level_2	4	uimsbf
user_nibble	4	uimsbf
user_nibble	4	uimsbf
}		
}		

The meaning of MH-content descriptor:

content_nibble_level_1 (genre 1): This field of 4 bits represents the first step classification of content identification. As for coding, it is provided separately. (Refer to ARIB STD-B10 Annex H.)

content_nibble_level_2 (genre 2): This field of 4 bits represents the second step classification of content identification. As for coding, it is provided separately. (Refer to ARIB STD-B10 Annex H.)

user_nibble (user genre): This field of 4 bits is defined by broadcasters.

7.4.3.22 MH-Parental Rating Descriptor

The MH-parantal rating descriptor represents viewing limit based on age, and is also used to extend based on the other control condition. The configuration of the MH-parental rating descriptor is shown in Table 7-54.

Table 7-54 MH-parental rating descriptor

Data strcture	Number of bit	Data notation
MH-Parental_Rating_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
country_code	24	bslbf
rating	8	uimsbf
}		
}		

The meaning of MH-parental rating descriptor:

country_code (country code): This field of 24 bits represents the name of country by using alphabetical three-letters code provided in ISO 3166-1. Each letter is coded by 8 bits according to ISO 8859-1, and is inserted to 24 bits field by the order.

Example: Japan is “JPN” for alphabetical three-letters code, and is coded as the following.

“0110 1010 0111 0000 0110 1110”

rating (rating by age control): This is a field of 8 bits, and represents recommended youngest age of viewer according to Table 7-55.

Table 7-55 Parental rating descriptor, age limit

rating	definition
0x00	undefined
0x01 – 0x0F	youngest age = rating + 3 years old
0x10 – 0xFF	defined by broadcaster

Ex: 0x04 represents the viewer is at least older than seven.

7.4.3.23 MH-Audio Component Descriptor

The MH-audio component descriptor represents each parameter of the audio elementary stream, and is also used to express the elementary stream by the letter format. The configuration of the MH-audio component descriptor is shown in Table 7-56.

Table 7-56 Configuration of MH-audio component descriptor

Data strcture	Number of bit	Data notation
MH-Audio_Component_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
reserved_future_use	4	bslbf
stream_content	4	uimsbf
component_type	8	uimsbf
component_tag	16	uimsbf
stream_type	8	uimsbf
simulcast_group_tag	8	bslbf
ES_multi_lingual_flag	1	bslbf
main_component_flag	1	bslbf
quality_indicator	2	bslbf
sampling_rate	3	uimsbf
reserved_future_use	1	bslbf
ISO_639_language_code	24	bslbf
if (ES_multi_lingual_flag == 1){		
ISO_639_language_code_2	24	bslbf
}		
for(i=0; i<N; i++){		
text_char	8	uimsbf
}		
}		

The meaning of MH-audio component descriptor:

stream_content (content of component): This field of 4 bits represents the kind of audio stream, and is coded according to Table 7-57.

Table 7-57 Content of component

Stream content	description
0x0	Reserved for use in the future
0x1	Not used
0x2	Sound stream that coding system is not specified
0x3	Sound stream by MPEG-4 AAC
0x4	Sound stream by MPEG-4 ALS
0x5	Not used
0x6 – 0x8	Reserved for use in the future
0x9	Not used
0xA – 0xB	Reserved for use in the future
0xC – 0xF	Defined by broadcasters

component_type (component type): This field of 8 bits represents the kind of audio component. Among 8 bits (b7 – b0), highest one bit (b7) represents presence/absence of the dialog control information, and is coded according to Table 7-58. The following two bits (b6 – b5) represent presence/absence of the audio for the hearing impared persons, and are coded according to Table 7-59. Lowest 5 bits (b4 – b0) represents the audio mode, and is coded according to Table 7-60.

Table 7-58 Dialog control (b7)

Dialog control b7	meaning
0	Audio stream does not include dialog control information.
1	Audio stream includes dialog control information.

Table 7-59 Audio for handicapped (b6 – b5)

Sound for the handicapped b6 – b5	meaning
00	Audio for the handicapped is not specified.
01	Audio explanation for the visual handicapped persons
10	Audio for the hearing impared persons
11	Reserved for use in the future

Table 7-60 Audio mode (b4 – b0)

Audio mode b4 – b0	meaning
0 0000	Reserved for use in the future
0 0001	1/0 mode (single mono)
0 0010	1/0＋1/0 mode (dual mono)
0 0011	2/0 mode (stereo)
0 0100	2/1 mode
0 0101	3/0 mode
0 0110	2/2 mode
0 0111	3/1 mode
0 1000	3/2 mode
0 1001	3/2＋LFE mode (3/2.1 mode [*1])
0 1010	3/3.1 mode [*1]
0 1011	2/0/0-2/0/2-0.1 mode [*1]
0 1100	5/2.1 mode [*1]
0 1101	3/2/2.1 mode [*1]
0 1110	2/0/0-3/0/2-0.1 mode [*1]
0 1111	0/2/0-3/0/2-0.1 mode [*1]
1 0000	2/0/0-3/2/3-0.2 mode [*1]
1 0001	3/3/3-5/2/3-3/0/0.2 mode [*1]
1 0010 – 1 1111	Reserved for use in the future

*1 Audio mode notation for multi-channel stereo: The number of channels is denoted as “upper layer (front/side/back) – middle layer (front/side/ back) – lower layer (front/side/ back).LFE”. But the layer which has no assigned channel is denoted as 0. Also, audio mode by only middle layer is denoted as “middle layer (front/side/ back).LFE”, and audio mode by only middle layer without side channel is denoted as “middle layer (front/ back).LFE”, for simplification.

component_tag (component tag): This is a field of 16 bits. The component tag is a label to identify component stream, and the value is the same as that of the component tag in the MH-stream identification descriptor (refer to the section 7.4.3.20). (Although in case that the MH-stream identification descriptor exists in MPT.)

stream_type (kind of stream type): This field of 8 bits represents the format of audio stream by stream_type provided in ISO/IEC 13818-1. It shall be 0x11 in the case of LATM/LOAS stream format, and it shall be 0x1C in the case of data stream format.

simulcast_group_tag (simulcast group identification): This field of 8 bits gives the same number to the component which is operating simulcast (transmitting the same content by different coding system). For the component which is not operating simulcast, it is set to 0xFF.

ES_multi_lingual_flag (ES multi-lingual flag): This flag of one bit is set to ‘1’ when two languages are multiplexed in ES (ES multi-lingual mode) for the 1/0+1/0 mode. In case of the other mode, it shall be undefined.

main_component_flag (main component flag): This flag of one bit shall be ‘1’ when the audio component is main audio. Also, in the case of the 1/0+1/0 mode, it shall be ‘1’ when the first audio component is main audio.

quality_indicator (audio quality indication): This field of two bits represents the audio quality mode, and is coded according to Table 7-61.

Table 7-61 Audio quality indicator

Quality indicator	description
00	Reserved for use in the future
01	Mode 1 *
10	Mode 2 *
11	Mode 3 *

*: For the details, see ARIB STD-B32 Part 2, Appendix, Chapter 2.

sampling_rate (sampling frequency): This field of three bits represents the sampling frequencies, and is coded according to Table 7-62.

Table 7-62 Sampling frequencies

Sampling frequency	Description
000	Reserved for use in the future
001	16 kHz
010	22.05kHz
011	24 kHz
100	Reserved
101	32 kHz
110	44.1 kHz
111	48 kHz

ISO_639_language_code (language code): This field of 24 bits represents the language of audio component. In the case of ES multi-lingual mode, it represents the language of the first audio component. The language code is represented as alphabetical three-letters code provided in ISO 639-2. Each letter is coded by 8 bits according to ISO 8859-1, and is inserted to 24 bits field by the order.

Example: Japanese language is “jpn” for alphabetical three-letters code, and is coded as the following.

“0110 1010 0111 0000 0110 1110”

ISO_639_language_code_2 (language code No.2): This field of 24 bits represents the language of the second audio component in the ES multi-lingual mode.

text_char (component description): This is a field of 8 bits. A series of character information field specifies the character description of the component stream.

7.4.3.24 MH-Target Region Descriptor

The MH-target region descriptor is used for describing the region which is the target for a program or a part of stream which composes a program. The configuration of MH-target region descriptor is shown in Table 7-63.

Table 7-63 Configuration of MH-target region descriptor

Data strcture	Number of bit	Data notation
MH-Target_Region_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
region_spec_type	8	uimsbf
target_region_spec ()		
}		

The meaning of MH-target region descriptor:

region_spec_type (region description system specification): This field of 8 bits specifies the system of region description in the following target_region_spec () structural body, and is coded according to Table 7-64.

Table 7-64 Designation of region description system

Value of the region description system specification	meaning
0x00	reserved
0x01	Designation of prefectural region for BS digital broadcasting
0x02 – 0xFF	reserved

target_region_spec () (region specifier): This represents data structure for specifying region which is provided for each region description system specification. (Refer to ARIB STD-B10 Annex G).

7.4.3.25 MH-Series Descriptor

MH-Series Descriptor is used to identify the series program. Configuration of MH-Series descriptor is shown in Table 7-65.

Table 7-65 Configuration of MH-series descriptor

Data strcture	Number of bit	Data notation
MH-Series_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
series_id	16	uimsbf
repeat_label	4	uimsbf
program_pattern	3	uimsbf
expire_date_valid_flag	1	uimsbf
expire_date	16	uimsbf
episode_number	12	uimsbf
last_episode_number	12	uimsbf
for(i=0; i<N; i++){		
series_name_char	8	uimsbf
}		
}		

The meaning of MH-series descriptor:

series_id (series identification): This is a field of 16 bits, and is an identifier to uniquely identify the series.

repeat_label (repeat broadcasting label): This field of 4 bits is used as a label to discriminate the program in case that the broadcasting period of a series and the re-broadcasting period of the series are piled up. 0x0 is given to the broadcasting for original series.

program_pattern (program pattern): This field of 3 bits represents program pattern of a series program according to Table 7-66. By this, it can be known when the event which belongs to the series appears next.

Table 7-66 Program pattern

Program pattern	description
0x0	Irregular (except the definition by 0x1 – 0x7)
0x1	A serialized program (every day, on weekday, only on Saturday and Sunday, etc.), Multiple programs in a week
0x2	Around weekly program
0x3	Around monthly program
0x4	Multiple programs on the same day
0x5	Division of long hours program
0x6	Regular or irregular program for storage
0x7	undefined

expire_date_valid_flag (expire date of validity flag): This flag of one bit represents that the value of the following expire_date is valid. In the case that the scheduled end date of the series is valid, it shall be '1'.

expire_date (validated date): This field of 16 bits represents the expiring date of the series as lower 16 bits of MJD. Even when the last event is not recognized on account of some reasons, the receiver recognizes as the series has ended after this date.

episode_number (episode number): This field of 12 bits represents the number of stories in the series of the program denoted by this descriptor. It can record from the first time to 4095th time. If the number of stories exceeds this, the series shall be defined separately. When the number of stories in the series of the program cannot be defined, it shall be 0x000.

last_episode_number (whole number of episodes): This field of 12 bits represents the whole number of stories for the concerned serial program. It can record from the first time to 4095th time. If the whole number of stories exceeds this, the series shall be defined separately. When the last time for the program is not decided, it shall be 0x000.

series_name_char (series name): In this field of letter codes, the series name is transmitted.

7.4.3.26 MH-SI Parameter Descriptor

The MH-SI parameter descriptor is used to represent transmission parameters of SI. The configuration of the MH-SI parameter descriptor is shown in Table 7-67.

Table 7-67 Configuration of MH-SI parameter descriptor

Data strcture	Number of bit	Data notation
MH-SI_Parameter_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
parameter_version	8	uimsbf
update_time	16	uimsbf
for(i=0; i<N; i++) {		
table_id	8	uimsbf
table_description_length	8	uimsbf
for(j=0; j<N; j++) {		
table_description_byte	8	uimsbf
}		
}		
}		

The meaning of MH-SI parameter descriptor:

parameter_version (parameter version): This is a field of 8 bits, and represents the update number for the transmission parameter of SI. Whenever the transmission parameter is updated, the value that is incrimented by one is recorded.

update_time (update time): This is a field of 16 bits, and the year, month and day when the recorded transmission parameter becomes valid are written by the lower 16 bits of MJD.

table_id (table identification): This field of 8 bits represents the kind of table which is described in the field of the following table description byte.

table_description_length (table description length): This field of 8 bits represents the byte length of the following table description byte.

table_description_byte (table description byte): This is a field of 8 bits. In a series of the table description regions, the transmission parameters for each kind of tables which are specified by operational provisions of broadcasters are described.

7.4.3.27 MH-Broadcaster Name Descriptor

The MH-broadcaster name descriptor describes the name of broadcaster. The configuration of the MH- broadcaster name descriptor is shown in Table 7-68.

Table 7-68 Configuration of MH-broadcaster name descriptor

Data strcture	Number of bit	Data notation
MH-Broadcaster_Name_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for(i=0; i<N; i++){		
char	8	uimsbf
}		
}		

The meaning of MH-broadcaster name descriptor:

char (character code): This is a field of 8 bits. A series of character information fields represents a broadcaster name.

7.4.3.28 MH-Service Descriptor

The MH-service descriptor represents the name of program channel and the broadcaster name with the service type by character codes. The configuration of the MH-service descriptor is shown in Table 7-69.

Table 7-69 Configuration of MH-service descriptor

Data strcture	Number of bit	Data notation
MH-Service_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
service_type	8	uimsbf
service_provider_name_length	8	uimsbf
for (i=0; i<N; i++) {		
char	8	uimsbf
}		
service_name_length	8	uimsbf
for (i=0; i<N; i++) {		
char	8	uimsbf
}		
}		

The meaning of MH-service descriptor:

service_type (service type): This is a field of 8 bits, and represents the kind of service according to the service types of Table 5-5.

service_provider_name_length (service provider name length): This field of 8 bits represents the byte length of the following broadcaster.

char (character code): This is a field of 8 bits. A series of character information fields represents the broadcaster name or the service name.

service_name_length (service name length): This field of 8 bits represents the byte length of the following service name.

7.4.3.29 IP Data Flow Descriptor

The IP data flow descriptor supplies the information of IP data flow which composes services. The IP data flow descriptor is arranged in MH-SDT, and is used for specifying the IP data flow which is involved in a TLV stream. In the circumstances where the IP packet is transmitted without using the TLV multiplex system, it becomes possible that the service identification related to the TLV stream identification is connected to the IP data flow. The configuration of the IP data flow descriptor is shown in Table 7-70.

Table 7-70 Configuration of IP data flow descriptor

Data strcture	Number of bit	Data notation
IP_Data_Flow_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
ip_version	1	bslbf
number_of_flow	7	uimsbf
for(i=0; i<N; i++) {		
ip_data_flow_id	8	uimsbf
if (ip_version=='0') /* IPv4 */		
{		
src_address_32	32	bslbf
dst_address_32	32	bslbf
}		
else if (ip_version=='1') /*IPv6*/		
{		
src_address_128	128	bslbf
dst_address_128	128	bslbf
}		
src_port	16	uimsbf
dst_port	16	uimsbf
}		
}		

The meaning of IP data flow descriptor:

ip_version (IP version): This represents the version of IP packets which are specified by descriptors. In the case of IPv4, it shall be '0', and in the case of IPv6, it shall be '1'.

number_of_flow (number of flow): This specifies the number of the IP flow to be recorded in this descriptor.

ip_data_flow_id (IP data flow ID): This shall be the label to discriminate the IP data flow in service. The IP data flow of which this ID is 0 represents that it transmits MPT.

src_address_32 (source IPv4 address): This represents the source IPv4 address of the IP data flow.

dst_address_32 (destination IPv4 address): This represents the destination IPv4 address of the IP data flow.

src_address_128 (source IPv6 address): This represents the source IPv6 address of the IP data flow.

dst_address_128 (destination IPv6 address): This represents the destination IPv6 address of the IP data flow.

src_port (source port number): This represents the source port number of the IP data flow.

dst_port (destination port number): This represents the destination port number of the IP data flow.

7.4.3.30 MH-CA Startup Descriptor

The MH-CA startup descriptor is provided in ARIB STD-B61. The MH-CA startup descriptor records startup informations for starting up the CAS program on CAS base. The configuration of the MH-CA startup descriptor is shown in Table 7-71.

Table 7-71 Configuration of MH-CA startup descriptor

Data strcture	Number of bit	Data notation
MH-CA_Startup_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
CA_system_ID	16	uimsbf
'111'	3	bslbf
CA_program_ID	13	uimsbf
2nd_load_flag	1	bslbf
load_indicator	7	bslbf
if (2nd_load_flag == '1') {		
'111'	3	bslbf
CA_program_ID	13	uimsbf
'1'	1	bslbf
load_indicator	7	bslbf
}		
exclusion_ID_num	8	uimsbf
for (i=0; i<exclusion_ID_num; i++){		
'111'	3	bslbf
exclusion_CA_program_ID	13	uimsbf
}		
load_security_info_len	8	uimsbf
for (i=0; i<load_security_info_len; i++) {		
load_security_info_byte	8	uimsbf
}		
for (i=0; i<N; i++) {		
private_data_byte	8	uimsbf
}		
}		

The meaning of MH-CA startup descriptor:

CA_system_ID (conditional access system identification): This represents the value of CA_system_ID corresponding to the CAS program to be started up.

CA_program_ID (CAS program identification): This represents the ID of the CAS program and the version number.

2nd_load_flag (second load flag): This represents the presence or absence of the indication of second load. When the second load flag is valid, the CAS program is indicated to load, in accordance with both conditions on the range indicated by the first load flag and the second load flag.

load_indicator (first/second load indication): This represents the indication for load (startup)

of the CAS program. The details are shown in Table 7-72. As for bit 3, the value recorded in the first load indicator shall be valid, and bit 3 in the second load indicator shall be ignored.

Table 7-72 Configuration of load_indicator

bit		meaning
bit6	-	reserved future use
bit5	-	reserved future use
bit4	-	reserved future use
bit3	0	In case that the range of bit0 - bit2 is specified (“before” “after”), the maximum version of the concerned range shall be the target of the CAS program. (Only first startup indication is valid.)
	1	In case that the range of bit0 - bit2 is specified (“before” “after”), the minimum version of the concerned range shall be the target of the CAS program. (Only first startup indication is valid.)
bit2	-	‘000’ : reserved ‘001’ : Version after specified CA_program_ID shall be the target to startup. ‘010’ : Version before specified CA_program_ID shall be the target to startup. ‘011’ : Only specified CA_program_ID shall be the target to startup. ‘100’ : private use ‘101’ - ‘111’ : reserved
bit1	-	
bit0	-	

exclusion_ID_num (number of exclusion CA_program_ID): This represents the number of the following exclusion CA_program_ID.

exclusion_CA_program_ID (exclusion CA_program_ID): This represents CA_program_ID which is excluded from execution.

load_security_info_len (load security information length): This represents the byte length of the following load security information.

load_security_info_byte (load security information): This represents the load security information. The body of receiver separates this information from the CA startup descriptor, and directly gives it to CAS base via API. Details of startup security information are not specified in this standard.

private_data_byte (private data): This represents private data. As for the private data, it is not specified in this standard.

7.4.3.31 MH-Data Component Descriptor

The MH-Data component descriptor is used for identifying the data coding system. For asset which is transmitted by using an application transmission system, the data_component_id is assigned by the MH-data component descriptor which is arranged in the second loop of the MP table. Additional identification information to be specified for each data coding system, which is used to describe the basic coding parameters of application, is not specified because the specification in MH-AIT is necessary and sufficient. The configuration of MH-data component descriptor is shown Table 7-73.

Table 7-73 MH-data component descriptor

Data strcture	Number of bit	Data notation
MH-Data_Component_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
data_component_id	16	uimsbf
for (i=0; i<N; i++) {		
additional_data_component_info	8	uimsbf
}		
}		

The meaning of MH-data component system descriptor:

data_component_id (data component identification): This field of 16 bits is used to identify the coding system of data. The assignment of this field value depends on the provision of standardization organizations. Refer to Table 7-74.

Table 7-74 Identification of data component system

data_component_id	meaning
0x0020	Closed-caption coding system for digital broadcasting (second generation)
0x0021	Multimedia coding system for digital broadcasting (second generation)

additional_data_component_info (additional identification information): This is a field of 8 bits, and is used for extening the number of identifier, or recording the supplemental information specified for each coding system. The data structure of information which is described in this region is separately provided for the data coding system.

7.4.3.32 MH-Local Time Offset Descriptor

The MH-local time offset descriptor is used in case that a fixed offset value is given between the actual time (UTC+9 hours) and the display time to human system when summer time is in effect. The configuration of the MH-local time offset descriptor is shown in Table 7-75.

Table 7-75 Configuration of MH-local time offset descriptor

Data strcture	Number of bit	Data notation
MH-Local_Time_Offset_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
country_code	24	bslbf
country_region_id	6	bslbf
reserved	1	bslbf
local_time_offset_polarity	1	bslbf
local_time_offset	16	bslbf
time_of_change	40	bslbf
next_time_offset	16	bslbf
}		
}		

The meaning of MH-local time offset descriptor:

country_code (country code): This field of 24 bits represents the name of country by using alphabetical three-letters code provided in ISO 3166-1. Each letter is coded by 8 bits according to ISO 8859-1, and is inserted to the field of 24 bits by the order.

Example: Japan is "JPN" for alphabetical three-letters code, and is coded as the following.

"0110 1010 0111 0000 0110 1110"

country_region_id (country region identification): This field of 6 bits is those for specifying region (zone) in the country. When the region is not discriminated, 000000 is used.

local_time_offset_polarity (local time offset polarity): This information of one bit specifies the following local time offset value and the polarity of the changed time offset value. When this bit is '0', it means to set offset time earlier than JST_time, and when it is '1', it means to set the offset time later than JST_time.

local_time_offset (local time offset): This field of 16 bits specifies present an offset time to JST_time in the range from -12 hours to +12 hours. In these 16 bits, the digit of 10 hours, the digit of one hour, the digit of 10 minutes and the digit of one minute of the offset time are coded by the binary coded decimal (BCD) of 4 bits, respectively.

time_of_change (time of change): This field of 40 bits includes date and time that will be changed to next different offset time which is represented by Japan Standard Time (UTC+9 hours) and Modified Julian Day (MJD). In this field, lower 16 bits of MJD are coded by 16 bits, and the following 24 bits are coded by six binary coded decimals (BCD's) of 4 bits.

next_time_offset (changed time offset): This field of 16 bits specifies an offset time after the date and time which is specified in time_of_change in the range from -12 hours to +12 hours. In these 16 bits, the digit of 10 hours, the digit of one hour, the digit of 10 minutes and the digit of one minute of offset time are coded by the binary coded decimal (BCD) of 4 bits, respectively.

7.4.3.33 MH-Component Group Descriptor

The MH-component group descriptor defines the combination of components in event and identifies it. The configuration of the MH-component group descriptor is shown in Table 7-76.

Table 7-76 Configuration of MH-component group descriptor

Data strcture	Number of bit	Data notation
MH-Component_Group_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
component_group_type	3	uimsbf
total_bit_rate_flag	1	uimsbf
num_of_group	4	uimsbf
for (i=0; i<num_of_group; i++) {		
component_group_id	4	uimsbf
num_of_CA_unit	4	uimsbf
for(j=0; j<num_of_CA_unit; j++) {		
CA_unit_id	4	uimsbf
num_of_component	4	uimsbf
for(k=0; k< num_of_component; k++) {		
component_tag	16	uimsbf
}		
}		
if (total_bit_rate_flag == 1) {		
total_bit_rate	8	uimsbf
}		
text_length	8	uimsbf
for (j=0; j<text_length; j++) {		
text_char	8	uimsbf
}		
}		
}		

The meaning of MH-component group:

component_group_type (component group type): This is a field of three bits, and represents the component group type according to Table 7-77.

Table 7-77 Component group type

Component group type	meaning
000	Multi-view TV service
000 – 111	undefined

total_bit_rate_flag (total bit rate flag): This is a flag of one bit, and represents the description state of total bitrate in the component group in event. When this bit is '0', it represents that total bitrate field in the component group does not exist in the concerned descriptor. When this bit is

'1', it represents that total bitrate field in the component group exists in the concerned descriptor.

num_of_group (number of group): This is a field of 4 bits, and represents the number of the component group in event.

component_group_id (component group identification): This is a field of 4 bits, and describes the component identification according to Table 7-78.

Table 7-78 Component group identification

Component group identification	meaning
0x0	Main group
0x0 – 0xF	Sub group

num_of_CA_unit (number of charge unit): This is a field of 4 bits, and represents the number of charge/non-charge unit in the component group.

CA_unit_id (charge unit identification): This is a field of 4 bits, and describes the charge unit identification to which component belongs, according to Table 7-79.

Table 7-79 Charge unit identification

Charge unit identification	description
0x0	Non-charge unit group
0x1	Charge unit group involving default ES group
0x2 – 0xF	Other charge unit group than the above-mentioned

num_of_component (number of component): This is a field of 4 bits, belongs to the concerned component group, and represents the number of component which belongs to charge/non-charge unit represented by immediately before CA_unit_id.

component_tag (component tag): This is a field of 16 bits, and represents the component tag value which belongs to the component group.

total_bit_rate (total bitrate): This is a field of 8 bits, and describes the total bitrate of component in the component group by rounding up the transmission rate of MMTP packet every 1Mbps.

text_length (component group description length): This is a field of 8 bits, and represents the byte length of the following component group description.

text_char (component group description): This is a field of 8 bits. A series of character information field describes explanation about the component group.

7.4.3.34 MH-Logo Transmission Descriptor

The MH-logo transmission descriptor is used for describing the character sequence for simple logo, pointing to logo by the CDT format, etc. The configuration of the MH-logo transmission descriptor is shown in Table 7-80.

Table 7-80 Configuration of MH-logo transmission descriptor

Data strcture	Number of bit	Data notation
MH-Logo_Transmission_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
logo_transmission_type	8	uimsbf
if (logo_transmission_type == 0x01) {		
reserved_future_use	7	bslbf
logo_id	9	uimsbf
reserved_future_use	4	bslbf
logo_version	12	uimsbf
download_data_id	16	uimsbf
for (i=0; i<N; i++) {		
logo_type	8	uimsbf
start_section_number	8	uimsbf
num_of_sections	8	uimsbf
}		
}		
else if (logo_transmission_type == 0x02) {		
reserved_future_use	7	bslbf
logo_id	9	uimsbf
}		
else if (logo_transmission_type == 0x03) {		
for (i=0; i<N; i++) {		
logo_char	8	uimsbf
}		
}		
else {		
for (i=0; i<N; i++) {		
reserved_future_use	8	bslbf
}		
}		
}		

The meaning of MH-logo transmission descriptor:

logo_transmission_type (logo transmission type): This field of 8 bits represents the transmission systems of logo shown in Table 7-81.

Table 7-81 Logo transmission system

logo_transmission_type	meaning
0x01	CDT transmission system 1: in case of directly referring to MH-CDT by download data identification
0x02	CDT transmission system 2: in case of indirectly referring to MH-CDT by download data identification, using logo identification
0x03	Simple logo system
0x04 – 0xFF	Reserved for use in the future

logo_id (logo identification): This field of 9 bits records ID of the logo data which is defined in the concerned service. (Refer to ARIB STD-B63, Annex 1)

logo_version (logo version number): This field of 12 bits records the version number of the concerned logo_id. (Refer to ARIB STD-B63, Annex 1)

download_data_id (download data identification): This field of 16 bits represents the identification of data to be downloaded. This agrees with the value of download_data_id of MH-CDT where logo data is arranged. Also refer to the section 7.3.3.10.

logo_type (logo type): This field of 8 bits represents the kind of logo data to be transmitted. About the value of logo type, refer to ARIB STD-B63, Annex 1.

start_section_number (start section number): This field of 8 bits represents the section number of the first section which stores and transmits the logo data of the concerned logo type.

num_of_sections (number of sections): This field of 8 bits represents the number of sections which store and transmit the logo data of the concerned logo type. When one logo data which is represented by a logo type is transmitted by dividing the logo data into multiple sections, it is transmitted by storing the divided data in successive sections which start from the section number for starting.

logo_char (character sequence for simple logo): This field of 8 bits records character sequence for simple logo.

7.4.3.35 MPU Extended Timestamp Descriptor

The MPU extended timestamp descriptor supplies the presentation time and the decoding time of access unit in MPU. The configuration of MPU extended timestamp descriptor is shown in Table 7-82. The MPU extended timestamp descriptor is arranged in the asset descriptor region of MPT.

Table 7-82 Configuration of MPU extended timestamp descriptor

Data strcture	Number of bit	Data notation
MPU_Extended_Timestamp_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
reserved	5	bslbf
pts_offset_type	2	uimsbf
timescale_flag	1	bslbf
if (timescale_flag == 1) {		
timescale	32	uimsbf
}		
if (pts_offset_type == 1) {		
default_pts_offset	16	uimsbf
}		
for (i=0; i<N; i++) {		
mpu_sequence_number	32	uimsbf
mpu_presentation_time_leap_indicator	2	uimsbf
reserved	6	bslbf
mpu_decoding_time_offset	16	uimsbf
num_of_au	8	uimsbf
for (j=0; j<num_of_au; j++} {		
dts_pts_offset	16	uimsbf
if (pts_offset_type == 2) {		
pts_offset	16	uimsbf
}		
}		
}		
}		

The meaning of MPU extended timestamp descriptor:

pts_offset_type (PTS offset type): When the difference of the presentation times between the previous access unit and the present access unit in the presentation order in the same MPU is the prescribed fixed value, this shall be 0, when the difference is specified as a default value by this descriptor, this shall be 1, and when the difference is individually specified for each access unit, it shall be 2.

timescale_flag (time scale flag): This represents whether the following time scale flag field exists or not, and when the time scale flag exists, this shall be '1', and when it does not exist, this shall be '0'.

timescale (time scale): This represents the unit of time in this descriptor. The value which one second divided by timescale shall be the unit of time.

default_pts_offset (default PTS offset): The difference of the presentation times between the previous access unit and the present access unit by the presentation order in the same MPU is represented by the unit of time which is represented by timescale, and it shall be the default value.

mpu_sequence_number (MPU sequence number): This represents the sequence number of MPU which involves the access unit that specifies decoding time, etc. in this descriptor.

mpu_presentation_time_leap_indicator (MPU presentation time leap indicator): When the leap second is adjusted to the system clock of transmission system, in the case that the MPU presentation time which the MPU sequence number is mpu_sequence_number is set based on the system clock from 9:00:00 am on the day before the leap second insertion day in Japanese time to the first system clock between 8:59:59 am and 9:00:00 am on the leap second insertion day, this represents 1. In the case that the MPU presentation time is set based on the system clock from 9:00:00 am on the day before the leap second deletion day to between 8:59:58 am and 8:59:59 am on the leap second deletion day, this represents 2. And in other cases, this represents 0. 3 shall be reserved. Also, the MPU presentation time is represented in the MPU timestamp descriptor.

mpu_decoding_time_offset (MPU decoding time offset): This represents the absolute value of the difference between the decoding time of first access unit by the decoding order and the presentation time of first access unit in the presentation order, by the unit of time represented in time scale.

num_of_au (number of access unit): This represents the number of access unit which specifies decoding time, etc. by this descriptor. Decoding time, etc. are stored in the decoding order of the access units.

dts_pts_offset (decoding time, presentation time offset): This represents the time from the decoding time to the presentation time of the access unit by the unit of time represented in time scale.

pts_offset (PTS offset): This represents the presentation interval of the access unit in MPU by the unit of time represented in time scale.

7.4.3.36 MPU Download Content Descriptor

The MPU download content descriptor is used for describing the attribute information of content which is downloaded by using MPU. The configuration of MPU download contents is shown in Table 7-83.

Table 7-83 Configuration of MPU download content descriptor

Data strcture	Number of bit	Data notation
MPU_Download_Content_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
reboot	1	bslbf
add_on	1	bslbf
compatibility_flag	1	bslbf
item_info_flag	1	bslbf
text_info_flag	1	bslbf
reserved	3	bslbf
component_size	32	uimsbf
download_id	32	uimsbf
time_out_value_DAM	32	uimsbf
leak_rate	22	uimsbf
reserved	2	bslbf
component_tag	16	uimsbf
if (compatibility_flag == 1) {		
compatibilityDescriptor ()		
}		
if (item_info_flag == 1) {		
num_of_items	16	uimsbf
for (i=0; i<num_of_items; i++) {		
item_id	32	uimsbf
item_size	32	uimsbf
item_info_length	8	uimsbf
for (j=0; j<item_info_length; j++) {		
item_info_byte	8	uimsbf
}		
}		
}		
private_data_length	8	uimsbf
for (i=0; i<N; i++) {		
private_data_byte	8	uimsbf
}		
if (text_info_flag == 1) {		
ISO_639_language_code	24	bslbf
text_length	8	uimsbf
for (i=0; i<text_length; i++) {		
text_char	8	uimsbf
}		
}		
}		

The meaning of MPU download content descriptor:

reboot (reboot): This designates whether it is necessary to reboot the receiver or not when

download is completed. If it is '1', it represents reboot, and if it is '0', it represents continuous working.

add_on (add on): This represents whether to rewrite existing item or to add item. If it is '1', it represents addition, and if it is '0', it represents rewriting.

compatibility_flag (compatibility flag): This is a flag which represents presence/absence of compatibilityDecriptor() in descriptor. This represents that if it is '1', compatibilityDecriptor() is coded, and it is not coded if it is '0'.

item_info_flag (item information flag): This is a flag which represents whether there is information for each item in descriptor or not. This represents that if it is '1', the information for each item is coded, and it is not coded if it is '0'.

text_info_flag (text information flag): This is a flag which represents whether there is service description at the end of descriptor or not. This represents that if it is '1', service description is coded, and it is not coded if it is '0'.

component_size (component size): This represents the sum of transmission data size (the unit is byte) in asset.

download_id (download identification): This specifies the download identification which is set in order to identify the reception number of this download. The download identification specified here is also specified to the data asset management table in actual delivery and to the extention header of the MMTP packet.

time_out_value_DAM (DAM time out value): The time out value to be recommended for receiving the DAM table of the concerned asset is represented by the unit of millisecond.

leak_rate (leak rate): The leak rate of transport buffer in the receiver is represented by the unit of 50 bytes/s.

component_tag (component tag): The value of the component tag for corresponding stream which is given by MH-stream identification descriptor of MPT is put in this field of 16 bits.

compatibilityDescriptor() (compatibility descriptor): An object for download such that cannot be specified by table_id_extention/group in MH-SDTT is specified here. When coding, an equivalent thing as compatibilityDescriptor is also put in the region of the descriptors in the data asset management table.

number_of_items (number of items): This represents the number of items.

item_id (item identification): This represents the item identification which is used for file transmission in download.

item_size (item size): This represents the byte length of the concerned item. 0 represents indefinite length.

item_info_length (item information length): This represents the byte length of item_info_byte.

item_info_byte (item information byte): Among the MH-type descriptor and the MH-Info descriptor which are described in the data asset management table, necessary descriptor is stored here.

private_data_length (private data length): This represents the byte length of private_data_byte.

private_data_byte (private data byte): The private data is recorded here. Usage of this region is decided in operation.

ISO_639_language_code (language code): The language in character description which is used for the service description is identified.

text_length (service description length): The size of service description is represented by the unit of byte.

text_char (service description): The information about service of transmitted download content is recorded here.

7.4.3.37 MH-Network Download Content Descriptor

The MH-network download content descriptor is used for describing the attribute information of downloaded contents by using a network. The configuration of the MH-network download content descriptor is shown in Table 7-84. This item is also provided in ARIB STD-B21.

Table 7-84 Configuration of MH-network download content descriptor

Data strcture	Number of bit	Data notation
MH-Network_Download_Content_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
reboot	1	bslbf
add_on	1	bslbf
compatibility_flag	1	bslbf
text_info_flag	1	bslbf
reserved	4	bslbf
component_size	32	uimsbf
session_protocol_number	8	uimsbf
session_id	32	uimsbf
retry	8	uimsbf
connect_timer	24	uimsbf
address_type	8	uimsbf
if (address_type == 0x00) {		
ipv4_address	32	uimsbf
port_number	16	uimsbf
}		
if (address_type == 0x01) {		
ipv6_address	128	uimsbf
port_number	16	uimsbf
}		
if (address_type == 0x02) {		
URL_length	8	uimsbf
for (i=0; i<URL_length; i++) {		
URL_byte	8	uimsbf
}		
}		
if (compatibility_flag == 1) {		
compatibilityDescriptor ()		
}		
private_data_length	8	uimsbf
for (i=0; i<N; i++) {		
private_data_byte	8	uimsbf

}		
if (text_info_flag == 1) {		
ISO_639_language_code	24	bslbf
text_length	8	uimsbf
for (i=0; i<text_length; i++) {		
text_char	8	uimsbf
}		
}		
}		

The meaning of MH-network download content descriptor:

reboot (reboot): This designates whether it is necessary to reboot the receiver or not when download is completed. If it is '1', it represents reboot, and if it is '0', it represents continuous working.

add_on (add on): This represents whether to rewrite existing item or to add item. If it is '1', it represents addition, and if it is '0', it represents rewriting.

compatibility_flag (compatibility flag): This is a flag which represents presence/absence of compatibilityDecriptor() in descriptor. This represents that if it is '1', compatibilityDecriptor() is coded, and it is not coded if it is '0'.

text_info_flag (text information flag): This is a flag which represents whether there is the service description at the end of descriptor or not. This represents that if it is '1', the service description is coded, and it is not coded if it is '0'.

component_size (component size): This represents the sum of transmission data size (the unit is byte) in asset.

session_protcol_number (session protocol number): This represents the protocol number of communication which is used in network download.

session_id (session identification): This represents the session identification of communication which is used in network download.

retry (retry): This represents the upper limit number of times when reconnecting to the communication server.

connect_timer (connection timer): This represents the connection delay parameter on which load dispersion of access to the communication server is supposed.

address_type (address type): This represents the kind of address information of the communication server, and follows the assignment in Table 7-85.

Table 7-85 Address type

address_type	meaning
0x00	This represents IPv4 address.
0x01	This represents IPv6 address.
0x02	This represents URL.

ipv4_address (IPv4 address): This represents IPv4 address of the communication server.

port_number (port number): This represents the port number of the communication server.

ipv6_address (IPv6 address): This represents IPv6 address of the communication server.

URL_length (URL length): This represents the byte length of URL of the communication server.

URL_byte (URL byte): This represents the character sequence of URL of the communication server.

compatibilityDescriptor () (compatibility descriptor): An object for download such that cannot be specified by table_id_extention/group in MH-SDTT is specified here. When coding, an equivalent thing as compatibilityDescriptor is also put in the region of descriptor of the data asset management table.

private_data_length (private data length): This represents the byte length of private_data_byte.

private_data_byte (private data byte): The private data is recorded here. The usage of this region is decided in operation.

ISO_639_language_code (language code): The language in character description which is used for the service description is identified here.

text_length (service description length): The size of service description is represented by the unit of byte.

text_char (service description): The information about service of transmitted download content is recorded here.

7.4.3.38 MH-Download Protection Descriptor

The MH-download protection descriptor is provided in ARIB STD-B61. The MH-download protection descriptor describes the location information of MMTP packet which transmits DCM and DMM, and the transmission information. When this descriptor is arranged in MH-SDTT, it represents the location information of DMM, and when this is arranged in the MP table, it represents the location information of DCM. The configuration of MH-download protection descriptor is shown in Table 7-86.

Table 7-86 Configuration of MH-download protection descriptor

Data strcture	Number of bit	Data notation
MH-DL_Protection_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
DL_system_ID	8	uimsbf
MMT_general_location_info ()		
encrypt_protocol_number	8	uimsbf
for (i=0; i<N; i++) {		
encrypt_info	8	bslbf
}		
}		

The meaning of MH-download protection descriptor:

DL_system_ID (download protection system identification): This represents the value of download protection system identification.

MMT_general_location_info () (location information): This represents the location of packet which transmits related information.

encrypt_protocol_number (encryption protocol number): This specifies encryption algorithm, etc. of related information or channel cipher. When this descriptor is arranged in MH-SDTT, it is applied to encrypt DCM and DMM, and when this is arranged in the MP table, it is applied to channel encryption.

encrypt_info (encryption information): This specifies an initial value, etc. of the encryption usage mode of related informations or channel encryption. When this descriptor is arranged in MH-SDTT, it is applied to encrypt DCM and DMM, and when this is arranged in the MP table, it is applied to channel encryption.

7.4.3.39 Application Service Descriptor

The application service descriptor describes entry informations, etc. of application related service. More than or equal to one application service descriptor is certainly arranged in MPT_descriptors_byte of the MP table related service with application. The configuration of the application service descriptor is shown in Table 7-87.

Table 7-87 Configuration of application service descriptor

Data strcture	Number of bit	Data notation
Application_Service_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
application_format	4	uimsbf
reserved_future_use	4	bslbf
document_resolution	4	bslbf
reserved_future_use	4	bslbf
default_AIT_flag	1	bslbf
DT_message_flag	1	bslbf
reserved_future_use	2	bslbf
EMT_num	4	uimsbf
AIT_location_info () {		
MMT_general_location_info ()		
}		
if (DT_message_flag == 1) {		
DT_message_location_info () {		
MMT_general_location_info ()		
}		
}		
for (j=0; j<EMT_num; j++) {		
EMT_tag	8	uimsbf
EMT_location_info () {		
MMT_general_location_info ()		
}		
}		
private_data ()		
}		

The meaning of application service descriptor:

application_format (application format): This represents application coding system which is applied in application service referred to by this descriptor. In this version, only 0x1: ARIB-HTML 5 is provided.

document_resolution (data content resolution): This field represents the resolution of the data content, and is coded according to Table 7-88.

Table 7-88 Data content resolution

value	meaning
0000	1920 x 1080
0001	3840 x 2160
0010	7680 x 4320
0011 – 1111	Reserved for use in the future

default_AIT_flag (default AIT flag): When this flag is '1', this represents that MH-AIT of the concerned delivery path is the default target of watching.

DT_message_flag (data transmission message flag): This represents whether data transmission message is involved in the concerned delivery path or not. If this flag is '1', it represents the data transmission message exists.

EMT_num (EMT number): This represents the number of EMT which is involved.

AIT_location_info (AIT location information): This represents the location which can acquire AIT by the format of MMT_general_location_info.

DT_message_location_info (data transmission message location information): This represents the location which can acquire the data transmission message by the format of MMT_general_location_info.

EMT_tag (EMT tag): This represents the identification information of EMT.

EMT_location_info (EMT location information): This represents the location which can acquire EMT by the format of MMT_general_location_info.

private_data (private data): This shall be an extension region which can be used in operation as required.

7.4.3.40 MH-Hierarchy Descriptor

The MH-hierarchy descriptor describes the information to identify the video stream component which is scalably coded. The MH-hierarchy descriptor is arranged in the descriptor region for each asset of the MP table. The configuration of MH-hierarchy descriptor is shown in Table 7-89.

Table 7-89 MH-hierarchy descriptor

Data strcture	Number of bit	Data notation
MH-Hierarchy_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
reserved	1	bslbf
temporal_scalability_flag	1	bslbf
spatial_scalability_flag	1	bslbf
quality_scalability_flag	1	bslbf
hierarchy_type	4	uimsbf
reserved	2	bslbf
hierarchy_layer_index	6	uimsbf
tref_present_flag	1	bslbf
reserved	1	bslbf
hierarchy_embedded_layer_index	6	uimsbf
reserved	2	bslbf
hierarchy_channel	6	uimsbf
}		

The meaning of MH-hierarchy descriptor:

temporal_scalability_flag (temporal scalable coding flag): This represents that if this flag is set to '0', the frame rate of bitstream of asset is improved which the concerned asset is referred to hierarchy_embedded_layer_index. '1' of this flag is the reserved value.

spatial_scalability_flag (spatial scalable coding flag): This represents that if this flag is set to '0', spatial resolution of bitstream of asset is improved which the concerned asset is referred to hierarchy_embedded_layer_index. '1' of this flag is the reserved value.

quality_scalability_flag (picture quality scalable coding flag): This represents that if this flag is set to '0', SNR quality or fidelity of bitstream of program asset is improved which the concerned asset is referred to hierarchy_embedded_layer_index. '1' of this flag is the reserved value.

hierarchy_type (kind of scalable coding): The relationship of hierarchy between the concerned hierarchy and basic hierarchy is defined in Table 7-90. When more than or equal to two kinds of scalable coding are applied, this field must be set to 8 (mixed scalable coding), and each flag of temporal_scalability_flag, spatial_scalability_flag and quality_scalability_flag must be set appropriately.

Table 7-90 Type of scalable coding

Hierarchy type	description
0	undefined
1	Spatial scalable coding
2	Picture quality scalable coding
3	Temporal scalable coding
4	Data partitioning
5	Extended bitstream
6	Private stream
7	Multi-view profile
8	Mixed scalable coding
9	MVC video sub-bitstream
10 - 14	undefined
15	Base layer, or MVC basic view point sub-bitstream, or AVC video sub-bitstream of MVC, or HEVC temporal sub-bitstream

hierarchy_layer_index (hierarchy index): This field defines the proper value of the concerned asset in the table of the coding hierarchy. These values must be unique in one MMT package.

tref_present_flag (TREF present flag): If this field is set to '0', it represents the possibility that there is the TREF (timestamp reference) field in the PES packet header of the concerned elementary stream. In the broadcasting system which uses MMT, this field is used by setting to '1'.

hierarchy_embedded_layer_index (basic hierarchy index): This field has the value of hierarchy_layer_index of basic asset, and must be accessed before decoding the asset which is connected by this scalable coding descriptor and must be displayed by the order of decoding. When the hierarchy_type is 15, it is not defined.

hierarchy_channel (hierarchy channel): This field represents the target channel number about the concerned asset among a series of transmission channel which has the order. About the whole definition of transmission hierarchy, most robust transmission channel is specified by the minimum value of this field.

(Note) There is a possibility that a specific hierarchy_channel is assigned to plural assets at the same time.

7.4.3.41 Content Copy Control Descriptor

The content copy control descriptor represents the information to control copy generation in a digital recording equipment for the whole service, and is used in order that a broadcaster (owner of copyright) conveys informations about copy or maximum transmission rate to the digital recording equipment. The configuration of the content copy control descriptor is shown in Table 7-91.

Table 7-91 Content copy control descriptor

Data strcture	Number of bit	Data notation
Content_Copy_Control_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
digital_recording_control_data	2	bslbf
maximum_bit_rate_flag	1	bslbf
component_control_flag	1	bslbf
reserved_future_use	4	bslbf
reserved_future_use	8	bslbf
if (maximum_bit_rate_flag == 1) {		
maximum_bitrate	8	uimsbf
}		
if (component_control_flag ==1) {		
component_control_length	8	uimsbf
for (i=0; i<N; i++) {		
component_tag	16	uimsbf
digital_recording_control_data	2	bslbf
maximum_bitrate_flag	1	bslbf
reserved_future_use	5	bslbf
reserved_future_use	8	bslbf
if (maximum_bitrate_flag == 1) {		
maximum_bitrate	8	uimsbf
}		
}		
}		
}		

The meaning of content copy control descriptor:

digital_recording_control_data (digital copy control information): This field represents the information to control copy generation, and it is coded according to Table 7-92.

Table 7-92 Control information of digital copy

Control information of digital copy	meaning
00	Able to be copied without restriction
01	Defined by broadcasters*1
10	Able to be copied for only one generation*2
11	Prohibited to copy

*1: Broadcaster can define freely.
*2: Received broadcasting signal can be recorded (copy of first generation), but the concerned signal which was recorded cannot be copied.

maximum_bit_rate_flag (maximum transmission rate flag): When this flag is '1', it represents that the following field with a maximum transmission rate exists. When it is '0', it represents that the following field with a maximum transmission rate does not exist.

component_control_flag (component control flag): This flag represents whether to specify the content copy control information for each component composing the program or not. When this flag is '1', it represents that the content copy control information is specified for each component which composes program. When this flag is '0', the content copy control information is specified for the whole program. In the case of transmitting this descriptor by MPT, '0' shall be set.

maximum_bitrate (maximum transmission rate): This field describes the transmission rate of MMTP packet for each program or component, by rounding up at each 1Mbps. In the case of a variable transmission rate, the maximum value is described.

component_control_length (component control length): This field represents the size of the following component control loop by the unit of byte.

component_tag (component tag): This field represents the label for identifying the asset of components which compose a program.

7.4.3.42 Content Usage Control Descriptor

The content usage control descriptor is used for representing the information about the copy control and the remote viewing control, in the case of storing broadcasting programs in hard disc, etc. and outputting videos and audio signals from the receiver. The configuration of the content usage control descriptor is shown in Table 7-93.

Table 7-93 Content usage control descriptor

Data strcture	Number of bit	Data notation
Content_Usage_Control_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
remote_view_mode	1	bslbf
copy_restriction_mode	1	bslbf
image_constraint_token	1	bslbf
reserved_future_use	5	bslbf
reserved_future_use	3	bslbf
retention_mode	1	bslbf
retention_state	3	bslbf
encryption_mode	1	bslbf
for (i=0; i<N; i++) {		
reserved_future_use	8	bslbf
}		
}		

The meaning of content usage control descriptor:

remote_view_mode (remote viewing mode): When this field is '1', it means that the remote viewing is possible. When it is '0', it means that the remote viewing is not possible.

copy_restriction_mode (copy restriction mode): This field represents the mode of restriction for the number of copies, and when this field is '1', it means the copy with the number limitation is possible. When it is '0', it means the copy with the number limitation is not possible.

image_constraint_token (resolution limitation bit): This field represents whether there is a limitation of picture quality for video signal output or not. When this field is '0', the resolution of video signal output must be limited. When it is '1', the limitation is not needed.

retention_mode (temporarily storing control bit): When this field is '0', it represents temporarily storing is possible even if digital_recording_control_data of the content copy control descriptor is "copy inhibit". When it is '1', temporarily storing is not possible.

retention_state (temporarily storing allowance time): This field represents the allowance time of temporarily storing after receiving a program, and it is coded according to Table 7-94.

Table 7-94 Time of retention

Retention time	meaning
111	One hour and a half
110	Three hours
101	Six hours
100	12 hours
011	One day
010	Two days
001	One week
000	Without limit

encryption_mode (output protection bit): This field represents whether there is an output protection of the IP interface output or not. When this field is '0', the IP interface output must be processed for protection. When it is '1', it may not be processed for protection.

7.4.3.43 Emergency News Descriptor

The emergency news descriptor represents an emergency flash news bulletin related to security and safety (emergency earthquake bulletin, news flash, breaking superimposition) is on the air. The configuration of the emergency news descriptor is shown in Table 7-95.

Table 7-95 Emergency news descriptor

Data strcture	Number of bit	Data notation
Emergency_News_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
transmit_timestamp	64	uimsbf
reserved_future_use	8	bslbf
}		

The meaning of emergency news descriptor:

transmit_timestamp (transmit timestamp): When an emergency news is broadcasted, the time when the emergency news descriptor is transmitted first is represented in the NTP long format. When the same emergency news continues, the same transmit timestamp shall be used. When the MSB of 32 bits denoting the unit of second is 0, the year 2036 shall be standard.

7.4.3.44 MH-CA Contract Information Descriptor

The MH-CA contract information descriptor is provided in ARIB STD-B61. The MH-CA contract information descriptor is used for confirming whether service or event can be reserved or not. The configuration of MH-CA contract information descriptor is shown in Table 7-96.

Table 7-96 Configuration of MH-CA contract information descriptor

Data strcture	Number of bit	Data notation
MH-CA_Contract_Info_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
CA_system_ID	16	uimsbf
CA_unit_id	4	uimsbf
num_of_component	4	uimsbf
for (i=0; i<num_of_component; i++) {		
component_tag	16	uimsbf
}		
contract_verification_info_length	8	uimsbf
for (i=0; i< contract_verification_info_length; i++) {		
contract_verification_info	8	uimsbf
}		
fee_name_length	8	uimsbf
for (i=0; i< fee_name_length; i++) {		
fee_name	8	uimsbf
}		
}		

The meaning of MH-CA contract information descriptor:

CA_system_ID (conditional access system identification): This represents the value of CA_system_ID corresponding to CAS program which startup.

CA_unit_ID (charge unit identification): This is a field of 4 bits, and represents the identification of charge unit/non-charge unit to which component belongs, according to Table 7-97. 0x0 is not used in this descriptor.

Table 7-97 Identification of charge unit

Charge unit identification	description
0x0	Non-charge unit group
0x1	Charge unit group involving asset which is primary in event.
0x2 – 0xF	Charge unit group other than the above-mentioned

num_of_component (number of component): This is a field of 4 bits, belongs to the concerned component group, and represents the number of component which belongs to charge/non-charge unit which is represented by immediately before CA_unit_id.

component_tag (component tag): This is a field of 16 bits, and represents the value of the component tag which belongs to the component group.

contract_verification_info_length (contract verification information length): This is a field of 8 bits, and represents the size of the following contract verification information by the unit of byte.

contract_verification_info (contract verification information): When this descriptor is arranged in MH-SDT, it is used for confirming if the service (or asset which composes service) can be reserved or not. For this previous confirmation, the receiver gives the contract verification information and the information of the date of viewing schedule to a CAS module, and a CAS module returns the result of judge about the possibility for viewing on the designated day.

fee_name_length (fee name information length): This is a field of 8 bits, and represents the size of the following fee name information by the unit of byte.

fee_name (fee name): This represents an explanation of the fee about described asset.

7.4.3.45 MH-CA Service Descriptor

The MH-CA service descriptor is provided in ARIB STD-B61. The MH-CA service descriptor represents the program channel of broadcaster who operates automatic display messages, and it describes display control informations of the concerned messages. The configuration of MH-CA service descriptor is shown in Table 7-98.

Table 7-98 Configuration of MH-CA service descriptor

Data strcture	Number of bit	Data notation
MH-CA_Service_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
CA_system_ID	16	uimsbf
ca_broadcaster_group_id	8	uimsbf
message_control	8	uimsbf
for (i=0; i<N; i++) {		
service_id	16	uimsbf
}		
}		

The meaning of MH-CA service descriptor:

CA_system_ID (conditional access system identification): This represents the value of CA_system_ID corresponding to the CAS program which starts up.

ca_broadcaster_group_id (broadcaster identification): This represents a code to identify the broadcaster that operates the conditional access system.

message_control (grace period): This represents the grace period by the day automatic display message which is previously implemented in a CAS module is displayed, by the unit of day. But 0xFF represents that the grace period is invalid (reservation of starting grace period).

service_id (service identification): This acts as a label to identify service.

7.4.3.46 Related Broadcaster Descriptor

The related broadcaster descriptor makes a connection with broadcasters of the other network in order to share NVRAM. The configuration of the related broadcaster descriptor is shown in Table 7-99.

Table 7-99 Configuration of related broadcaster descriptor

Data strcture	Number of bit	Data notation
Related_Broadcaster_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
num_of_broadcaster_id	4	uimsbf
num_of_affiliation_id	4	uimsbf
num_of_original_network_id	4	uimsbf
reserved	4	bslbf
for (i=0; i<num_of_broadcaster_id; i++) {		
network_id	16	uimsbf
broadcaster_id	8	uimsbf
}		
for (i=0; i<num_of_affiliation_id; i++) {		
affiliation_id	8	uimsbf
}		
for (i=0; i<num_of_original_network_id; i++) {		
original_network_id	16	uimsbf
}		
}		

The meaning of related broadcaster descriptor:

num_of_broadcaster_id (number of broadcaster identification): This is a field of 4 bits, and represents the number of sets of the subsequent network identification and broadcaster identification.

num_of_affiliation_id (number of affiliation identification): This is a field of 4 bits, and represents the number of the subsequent affiliation identification.

num_of_original_network_id (number of original network identification): This is a field of 4 bits, and represents the number of the network identification in the subsequent original delivery system.

network_id (network identification): This is a field of 16 bits, and identifies the networks of broadcasters which share NVRAM.

broadcaster_id (broadcaster identification): This is a field of 8 bits, and identifies broadcasters in the networks.

affiliation_id (affiliation identification): This is a field of 8 bits, and identifies the affiliations of digital terrestrial broadcasting which shares NVRAM.

original_network_id (original network identification): This is a field of 16 bits, and identifies the networks of digital terrestrial broadcasting which shares NVRAM.

7.4.3.47 Multimedia Service Information Descriptor

The multimedia service information descriptor is used for describing the detailed information of each content in multimedia services. The configuration of multimedia service information descriptor is shown in Table 7-100.

Table 7-100 Configuration of multimedia service information descriptor

Data strcture	Number of bit	Data notation
Multimedia_Service_Info_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
data_component_id	16	uimsbf
if (data_component_id == 0x0020) {		
component_tag	16	uimsbf
ISO_639_language_code	24	bslbf
text_length	8	uimsbf
for (i=0; i<N; i++) {		
text_char	8	uimsbf
}		
}		
if (data_component_id == 0x0021) {		
associated_contents_flag	1	bslbf
reserved	7	bslbf
}		
selector_length	8	uimsbf
for(i=0; i<N; i++){		
selector_byte	8	uimsbf
}		
}		

The meaning of multimedia service information descriptor:

data_component_id (data coding system identification): This is a field of 16 bits, and describes the same value as the data coding system identification of the MH-data component descriptor.

component_tag (component tag): This is a field of 16 bits. The component tag is a label to identify component streams, and is the same value as the component tag in the MH-stream identifier descriptor (refer to Section 7.4.3.20). (Only in the case that the MH-stream identifier descriptor exists in MPT.)

ISO_639_language_code (language code): This is a field of 24 bits, and the language of the character description used in the subsequent service description is represented as the alphabetical three-letters code provided in ISO 639-2.

text_length (content description length): This is a field of 8 bits, and represents the byte length of the subsequent content description.

text_char (content descrition): This is a field of 8 bits, and describes explanation about transmitted content.

associated_contents_flag (associated contents flag): This is a field of one bit, and when the

multimedia service which is described by this descriptor is the additional data service for television programs, it represents that the service involves actually associated contents to the television program. For the service which is not additional data service, it shall be always '0'.

selector_length (selector length): This is a field of 8 bits, and represents the byte length of the subsequent selector region.

selector_byte (selector byte): This is a field of 8 bits, and describes necessary information to acquire data. The data structure of the information described in this region is provided for each data coding system in operation.

Chapter 8: Transmission of Video Signal and Audio Signal

8.1 Transmission of Video Signal

8.1.1 Summary of Packetization of Video Signal

The interface between HEVC which is a video coding system and MMT which is a media transport system is NAL (Network Abstraction Layer) unit. For transmission of video signal which is coded by HEVC, NAL unit is transmitted as MFU. VPS, SPS and PPS which are necessary information to decode HEVC bitstream shall be MFU as NAL unit, respectively. So, hev1 is used as asset type to identify asset of HEVC bitstream. In case that the output of HEVC encoder is bytestream, by eliminating byte start code, the size of NAL unit by the unit of byte represented by 32 bits (integer without sign) as information of length is added to immediately before NAL unit and it shall be MFU.

Overview of the processing that MPU is configured from NAL unit which HEVC encoder outputs, furthermore MPU is MMTP packetized is shown in Fig. 8-1. The size of MPU is related to the delay until outputting video signal in switching channel of broadcasting in the receiver. So, MPU is configured by the interval of IRAP in video coding.

MMTP packet consists of MMTP packet header and MMTP payload part. In MMTP payload, plural NAL units can be stored. Also, in case that the size of NAL unit is bigger, comparing the size of IP packet which can be transmitted, NAL unit is divided and stored in payload. As shown in Fig. 8-1, the size of non-VCL NAL unit is generally smaller than NAL unit of slice segment, so plural non-VCL NAL units are collected to configure one MMTP payload. Whereas, NAL unit of slice segment is divided into plural MMTP payloads.

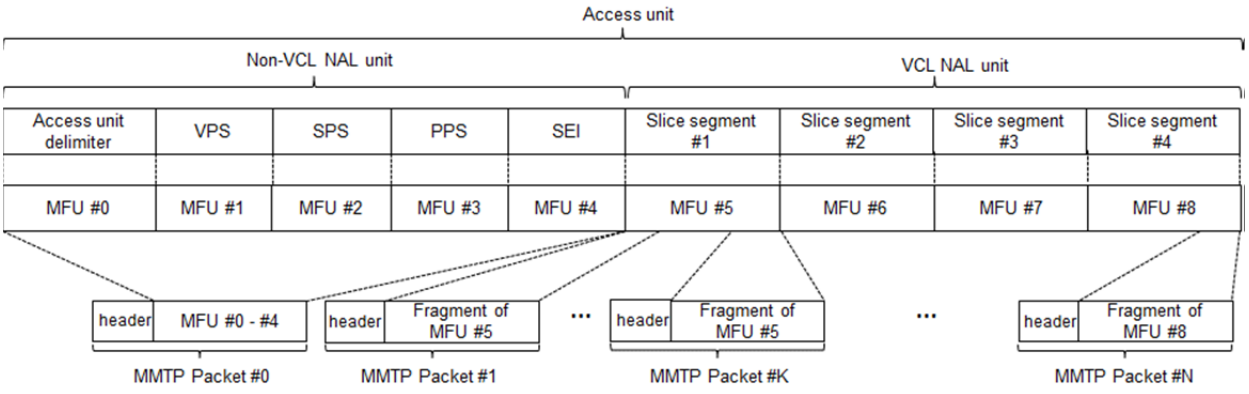


Fig. 8-1 Overview of NAL unit configuring MPU and MMTP packetization

By using information of MFU involved in MMTP payload, the receiver can detect access unit and the head of slice segment.

8.1.2 Transmission of Temporal Scalable Coding Stream

Overview about transmission of temporal scalable coding HEVC bitstream is shown in Fig. 8-2. When configuring package, HEVC temporal sub-bitstream for 60Hz decoding display and subset for 120Hz decoding display shall be separate assets. In Fig. 8-2, the former as asset 1, the latter as asset 2 are shown as an example. As they are separate assets, access unit of asset 1 and access unit of asset 2 are transmitted by MMTP packet of separate packet ID

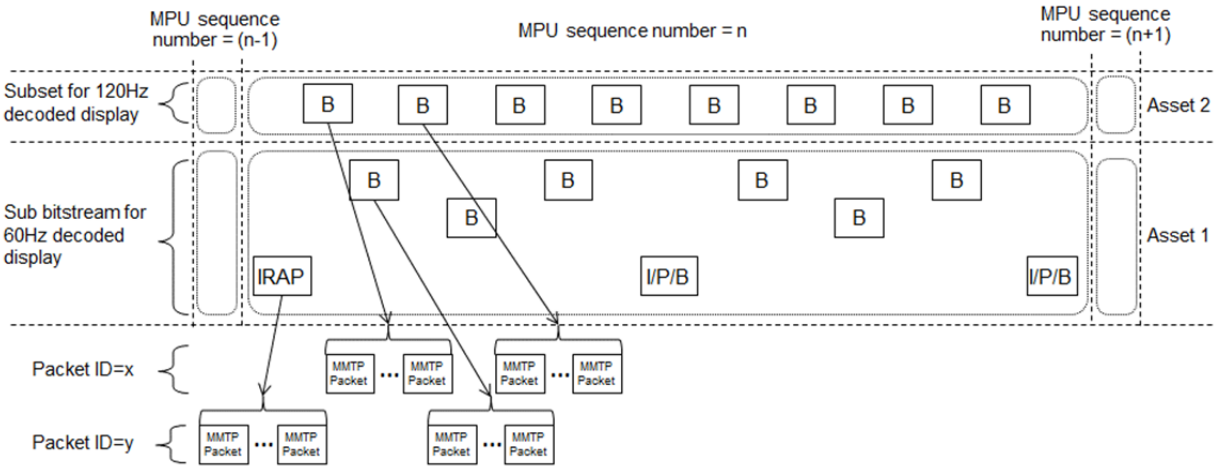


Fig. 8-2 Overview of transmission of temporal scalable coding HEVC bitstream

Also, MPU involving access unit of subset for 120Hz decoding display shall be added by the same MPU sequence number as MPU involving access unit of sub-bitstream for 60Hz decoding diplay of GOP which these access units belong to. By giving the same MPU sequence number to MPU's which are related to time, it makes possible for the receving terminal to easily identify that these assets involve access units which belong to the same GOP.

In an example of Fig. 8-2, asset 2 depends upon asset 1 for decoding. So, dependency descriptor is inserted into descriptor region which describes information of asset 2 in MP table, and asset ID of asset 1 is described for dependency destination. Also, MPU timestamp descriptor and MPU extended timestamp descriptor shall be added to both asset 1 and asset 2.

Network and packet ID which asset 1 and asset 2 are transmitted shall be shown for each asset by location information of MP table.

8.2 Transmission of Audio Signal

8.2.1 Summary of Transmission of Audio Signal

The interface between MPEG-4 AAC or MPEG-4 ALS which is an audio coding system and MMT which is a media transport system is LATM/LOAS stream format (sequence of AudioSyncStream ()) or data stream format. LATM (Low Overhead MPEG-4 Audio Transport Multiplex) involves channel configuration information of audio data, and in addition, supplies multiplexing function of order, connection, etc. of audio data. Also, LOAS (Low Overhead Audio Stream) supllies synchronization function.

As for transmission of audio signal of LATM/LOAS stream format, AudioMuxElement () that synchronization byte and length information are removed from AudioSyncStream() is

transmitted as MFU. In the receiver, it is output to audio decoder as AudioSyncStream() that synchronization byte and length information are added to AudioMuxElement() which is involved in the received MFU.

Also, for transmission of audio signal by data stream format, Raw Dara Stream provided in ARIB STD-B32 Part 2 shall be MFU. For all, processing of transmission in MMTP packet of MFU is the same as the transmission of video signal.

Chapter 9: Transmission of Closed-Caption and Superimposition

9.1 Summary of Closed-Caption and Superimposition

In closed-caption and superimposition transmission system, closed-caption or superimposition of one language is transmitted by one asset. So, in transmitting multi-lingual closed-caption and superimposition, they are transmitted by using multiple assets. An example of transmitting closed-caption and superimposition of two languages respectively is shown in Fig. 9-1.

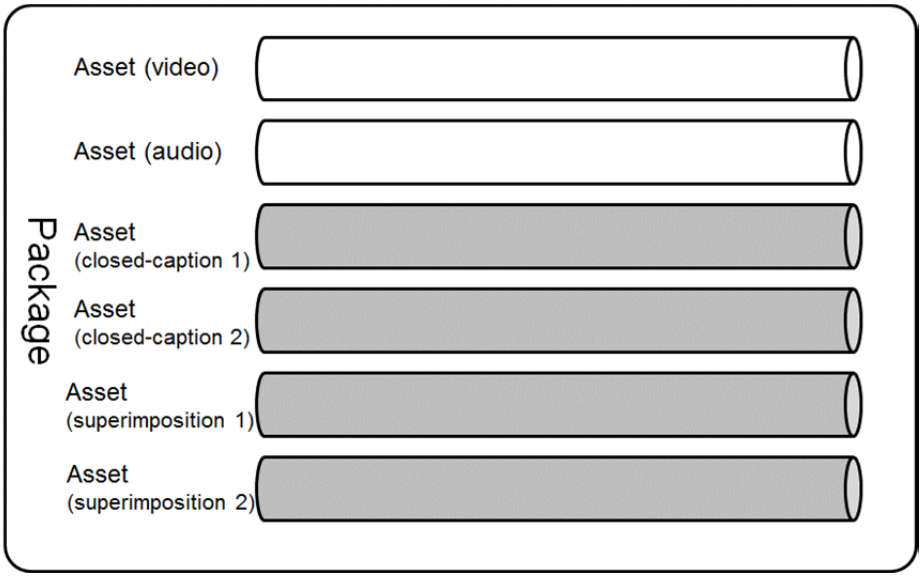


Fig. 9-1 An example of transmission of closed-caption and superimposition

Control information about transmission of closed-caption and superimposition is transmitted by MP table, and a series of file which involves TTML document file presented in the range of a certain time is capsulized in synchronous type MPU, and is transmitted. Summary of acquisition of closed-caption and superimposition is shown in Fig. 9-2.

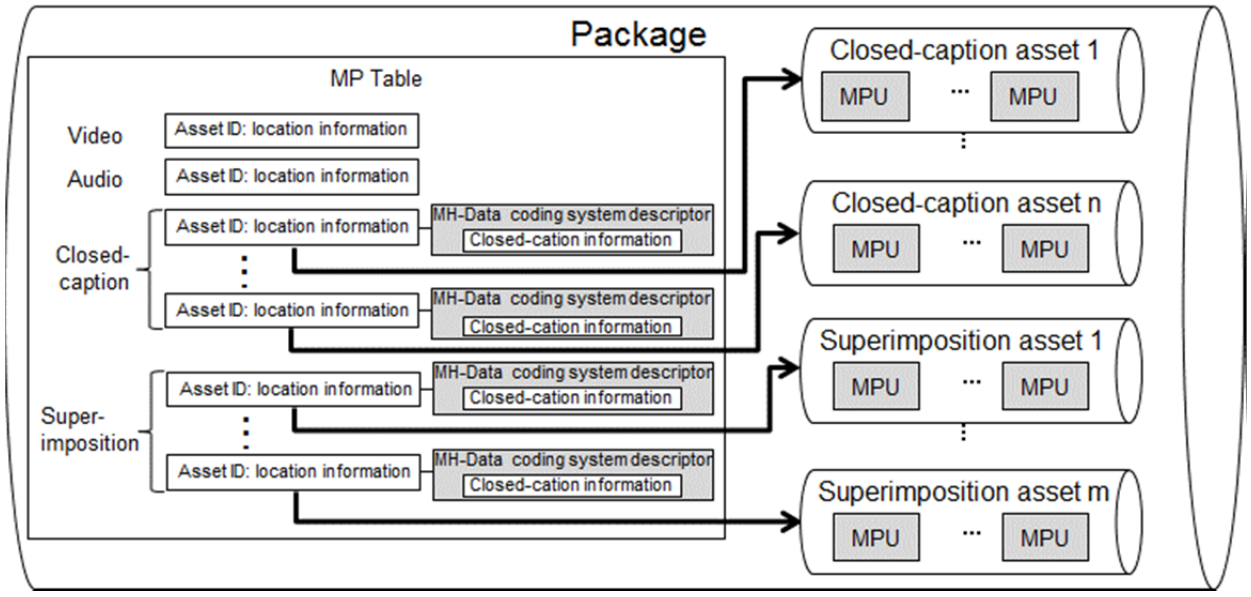


Fig. 9-2 Summary of acquisition of closed-caption and superimposition

The receiver refers to MP table, and detects whether there are asset or not which transmits closed-caption and superimposition. In case that closed-caption and superimposition are transmitted, the asset whose asset type is stpp exists, and MH-data component descriptor is arranged in the asset descriptor region. Additional information about closed-caption and superimposition is described in MH-data component descriptor. By referring to MP table, location information of asset is identified, and packet ID of asset which transmits closed-caption and superimposition is specified. By expanding data in MPU which is transmitted in asset of specified packet ID, the file which is transmitted as closed-caption and superimposition can be acquired.

9.2 Transmission of Closed-Caption and Superimposition

Closed-caption and superimposition are transmitted by using synchronous type MPU (MPU with Timed Media Data) which is defined in MMT standard.

9.2.1 Configuration of MPU/MFU for Closed-Caption and Superimposition

In process of closed-caption and superimposition, MPU becomes the unit of processing. MPU involves data of TTML document file, video file, audio file and external character file which are presented in a certain time interval, and is the unit that presentation of closed-caption can be processed by the unit of MPU. Configuration of MPU/MFU for closed-caption and superimposition is shown in Fig. 9-3.

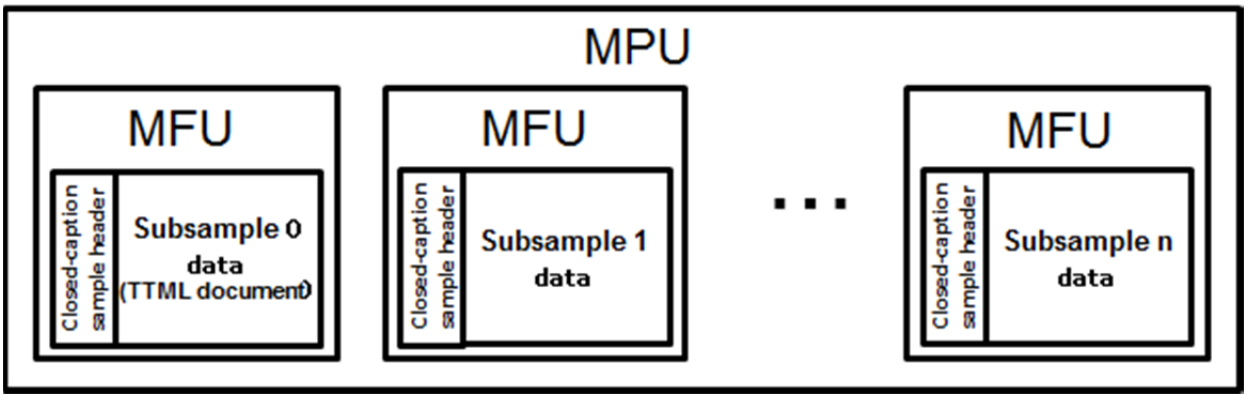


Fig. 9-3 Configuration of MPU/MFU for closed-caption and superimposition

MPU for closed-caption and superimposition does not arrange MPU metadata and movie fragment data, and is composed of more than or equal to one MFU. MFU is composed of sample header part of closed-caption and subsample data part. TTML document is always arranged in the first MFU in MPU, and in case that there are some external resource to which TTML document refers, the file of external resource is arranged as subsample in the following MFU. TTML document, video file, audio file and external character file are stored as subsamples of one MFU, respectively.

An example of configuratuion of MPU for closed-caption and superimposition is shown in Fig. 9-4. In this example, TTML document refer to two video files and one font file.

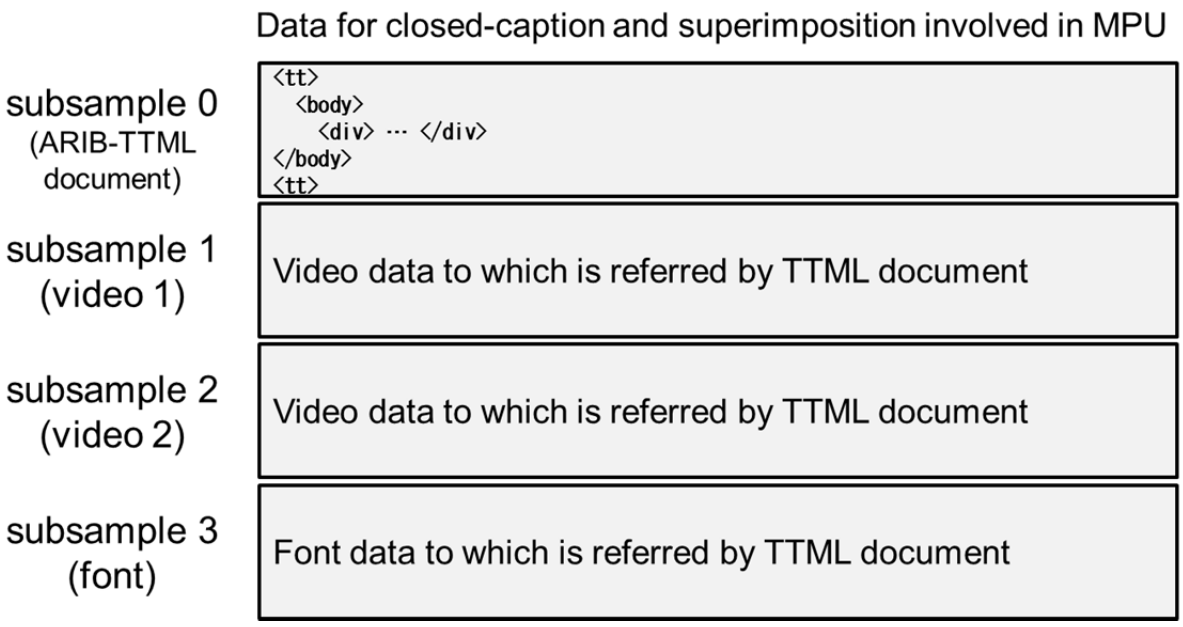


Fig. 9-4 Configuration of MPU for closed-caption and superimposition

9.2.2 Configuration of MFU for Closed-Caption and Superimposition

The configuration of MFU for closed-caption and superimposition is shown in Table 9-1.

Table 9-1 Configuration of MFU for closed-caption and superimposition

Data atructure	Number of bit	Data notation
MFU_data_byte (){		
subtitle_tag	8	uimsbf
subtitle_sequence_number	8	uimsbf
subsample_number	8	uimsbf
last_subsample_number	8	uimsbf
data_type	4	bslbf
length_extension_flag	1	bslbf
subsample_info_list_flag	1	bslbf
reserved	2	bslbf
if (length_extension_flag == 1) {		
data_size	32	uimsbf
} else {		
data_size	16	uimsbf
}		
if (subsample_number == 0 &&		
last_subsample_number > 0 &&		
subsample_info_list_flag == 1) {		
for (i=1; i<last_subsample_number+1; i++) {		
subsample_i_data_type	4	bslbf
reserved	4	bslbf
if (length_extension_flag == 1) {		
subsample_i_data_size	32	uimsbf
} else {		
subsample_i_data_size	16	uimsbf
}		
}		
}		
for (j=0; j<N; j++) {		
data_byte	8	bslbf
}		
}		

The meaning of MFU of closed-caption and superimposition:

subtitle_tag (closed-caption identification tag): This is the label to identify stream of closed-caption and superimposition.

subtitle_sequence_number (closed-caption sequence number): This represents the sequence number of the whole closed-caption data to be transmitted by the concerned closed-caption asset. This is incremented between successive MPU's.

subsample_number (subsample number): Number of closed-caption data (subsample) in the concerned MPU is represented by ascending order. In the first MFU in the concerned MPU, number of transmitted closed-caption data is 0.

last_subsample_number (last subsample number): This represents the last subsample number of closed-caption data which is transmitted by the concerned MPU.

data_type (data type): Data type of closed-caption data which is transmitted by this MFU is

coded according to Table 9-2.

Table 9-2 Meaning of data type

Value of data_type	meaning
0000	TTML document file
0001	Video file by PNG format
0010	Video file by SVG format
0011	Audio file by PCM (AIFF-C) format
0100	Audio file by MP3 format
0101	Audio file by MPEG-4 AAC format
0110	Font file by SVG format
0111	Font file by WOFF format
1000-1111	Reserved for use in the future

length_extension_flag (length information extension flag): When message data length field which represents the size of message is made to be 32 bits, this shall be '1', and when it is made to be 16 bits, this shall be '0'.

subsample_info_list_flag (whole closed-caption information flag): In case of describing hint information related to all closed-caption data which is transmitted by the concerned MPU, this shall be '1', and in case of not describing hint information, this shall be '0'.

subsample_i_data_type (subsample i data type): This represents data type of closed-caption data which is transmitted by i'th MFU, and is coded according to Table 9-2.

subsample_i_data_size (subsample i data length): This represents the size of closed-caption data which is transmitted by i'th MFU.

data_byte (data byte): Data byte of closed-caption data of the concerned MFU is stored in this.

9.3 Descriptor in Transmission of Closed-Caption and Superimposition

About closed-caption and superimposition which are transmitted by using closed-caption and superimposition transmission system, MH-data coding system descriptor is arranged in MP table and data_component_id is assigned. Moreover, in additional_data_component_info field which is specified by each coding system of MH-data coding system descriptor, additional arib subtitle information which has the data structure shown in Table 9-3 is stored.

Table 9-3 Configuration of additional arib subtitle information for transmission system of closed-caption and superimposition

Data structure	Number of bit	Data notation
Additional_Arib_Subtitle_Info () {		
subtitle_tag	8	uimsbf
subtitle_info_version	4	uimsbf
start_mpu_sequence_number_flag	1	bslbf
reserved	3	bslbf
ISO_639_language_code	24	uimsbf
type	2	bslbf
subtitle_format	4	bslbf
OPM	2	bslbf
TMD	4	bslbf
DMF	4	bslbf
resolution	4	bslbf
compression_type	4	bslbf
if (start_mpu_sequence_number_flag == 1) {		
start_mpu_sequence_number	32	uimsbf
}		
if (TMD == 0010) {		
reference_start_time	64	uimsbf
reference_start_time_leap_indicator	2	uimsbf
reserved	6	bslbf
}		
}		

The meaning of additional arib subtitle information of closed-caption and superimposition transmission system:

subtitle_tag (closed-caption identification tag): This is the label to identify closed-caption and superimposition stream.

subtitle_info_version (closed-caption information version): This represents the version of additional arib subtitle information of closed-caption and superimposition.

start_mpu_sequence_number_flag (start MPU sequence number flag): When the following start_mpu_sequence_number field is arranged, this shall be '1', and when it is not arranged, this shall be '0'.

ISO_639_language_code (language code): This field of 24 bits represents the language code for the language identified by closed-caption identification tag as alphabetical three-letters code provided in ISO 639-2. Each letter is coded by 8 bits according to ISO 8859-1, and is inserted to 24 bits field by the order.

Example: Japanese language is "jpn" for alphabetical three-letters code, and is coded as the following.

"0110 1010 0111 0000 0110 1110"

type (type of closed-caption): This represents the classification of closed-caption and superimposition according to Table 9-4.

Table 9-4 Meaning of closed-caption type

Value of closed-caption type	meaning
00	Closed-caption
01	Superimposition
10	Reserved for use in the future
11	Reserved for use in the future

subtitle_format (closed-caption description system identification): This specifies description system of closed-caption and superimposition, involving version and profile. The meaning of closed-caption description system identification is shown in Table 9-5.

Table 9-5 Meaning of identification for closed-caption description system

Value of closed-caption description system identification	meaning
0000	ARIB-TTML description system which is identified in the following name space http://www.arib.or.jp/ns/arib-ttml/v1_0
0001 – 1111	Reserved for use in the future

OPM (operation mode): This represents operation mode of the whole system involving transmission of closed-caption and coding. The meaning of operation mode is shown in Table 9-6.

Table 9-6 Meaning of operation mode

Value of operation mode	Name of operation mode	meaning
00	Live mode	Operation mode by which TTML document is updated in a short time. As a result, the dependency relationship between documents can occur before and after uptate. The same TTML documents are transmitted only once. It is supposed to be applied to live program.
01	Segment mode	Operation mode by which TTML document is divided into plural documents which can be operated independently and updated. Operation mode by which operation is possible, as there is no dependency relationship between TTML documents before and after update. The same TTML documents are transmitted only once.
10	Program mode	Operation mode by which the same TTML documents are transmitted repeatedly as TTML document which can be operated independently. The operation is supposed that TTML document by the unit of program is transmitted.
11		Reserved for use in the future

TMD (time control mode): This represents time control mode at receiving and reproducing, and specifies the presentation time by MPU time stamp which is added to MPU, time code which is described in TTML document or the other information. The meaning of time control mode is shown in Table 9-7.

Table 9-7 Meaning of time control mode

Value of time control mode	Time control system	meaning
0000	TTML description (UTC)	Presentation time is represented by time code in TTML document as UTC.
0001	TTML description (MH-EIT starttime for the starting point)	Time is represented by time code in TTML document at starttime of MH-EIT as the starting point.
0010	TTML description (reference starttime for the starting point)	Time is represented by time code in TTML document at reference starttime of this descriptor as the starting point.
0011	TTML description (MPU timestamp for the starting point)	Time is represented by time code in TTML document at MPU presentation time of MPU timestamp as the starting point.
0100	TTML description (NPT)	This represents the time that time code in TTML document is equivalent to NPT based on UCT-NPT reference descriptor which is transmitted by event message table (EMT).
1000	MPU Timestamp	Presentation time is represented by only MPU presentation time of MPU timestamp descriptor, not following time code in TTML document.
1111	without time control	This is used in such a case of presenting closed-caption and superimposition promptly, following neither MPU timestamp descriptor nor time code in TTML document.

DMF (display mode): Display mode of closed-caption sentence is represented by two bits of presentation movement at receiving and reproducing, respectively. The meaning of display mode is shown in Table 9-8.

Table 9-8 Meaning of display mode

Value of display mode		meaning
b4 b3	b2 b1	
0 0		Automatic display when received
0 1		Automatic non-display when received
1 0		Selective display when received
1 1		Reserved for use in the future
	0 0	Automatic display when playback
	0 1	Automatic non-display when playback
	1 0	Selective display when playback
	1 1	Reserved for use in the future

resolution (display resolution): This represents initial state of display resolution of the screen to be displayed by closed-caption. The meaning of display resolution is shown in Table 9-9.

Table 9-9 Meaning of display resolution

Value of display resolution	meaning
0000	1920 x 1080
0001	3840 x 2160
0010	7680 x 4320
0011 – 1111	Reserved for use in the future

compression_type (compression type): This represents compression types of closed-caption data. Closed-caption data for the target of compression is only TTML document which is arranged first among transmitted sample data. The meaning of compression types is shown in Table 9-10.

Table 9-10 Meaning of compression

Value of copression	meaning
0000	uncompression
0001	EXI(with schema specification)
0010	EXI(without schema specification)
0011 – 1111	Reserved for use in the future

start_mpu_sequence_number (start MPU sequence number): This represents the first MPU sequence number of closed-caption and superimposition for which setting by this descriptor is valid.

reference_start_time (reference start time): UTC time which is the starting point of time code in TTML document in case that TMD is 0010 is represented by NTP long format.

reference_start_time_leap_indicator (reference start time leap second indicator): When leap second is adjusted fot system clock of the transmission system, if reference start time is between am 9:00:00 on the day before leap second insertion day and first am 8:59:59 on the leap second insertion day of Jappan time, this represents 1. And if it is between am 9:00:00 on the day before leap second deletion day and am 8:59:59 on the leap second deletion day, this represents 2, and in other case, this represents 0. 3 shall be reserved.

Chapter 10: Transmission of Application

10.1 Summary of Application Transmission System

This chapter provides the transmission system equivalent to the data carourcel transmission system which is provided by ARIB STD-B24.

For the download of data to the receiver and the transmission of contents in multimedia services, etc., by transmitting data repeatedly, the receiver acquires necessary data at an arbitrary point in the air time. The data is capsulized in asynchronous type MPU by the unit of item (file) and transmitted. The control information about the transmission of application is transmitted by using the data transmission message. In the data transmission message, only one among the data directory management table, the data asset management table and the data content management table is stored. The overview of acquiring file that configures application is shown in Fig. 10-1.

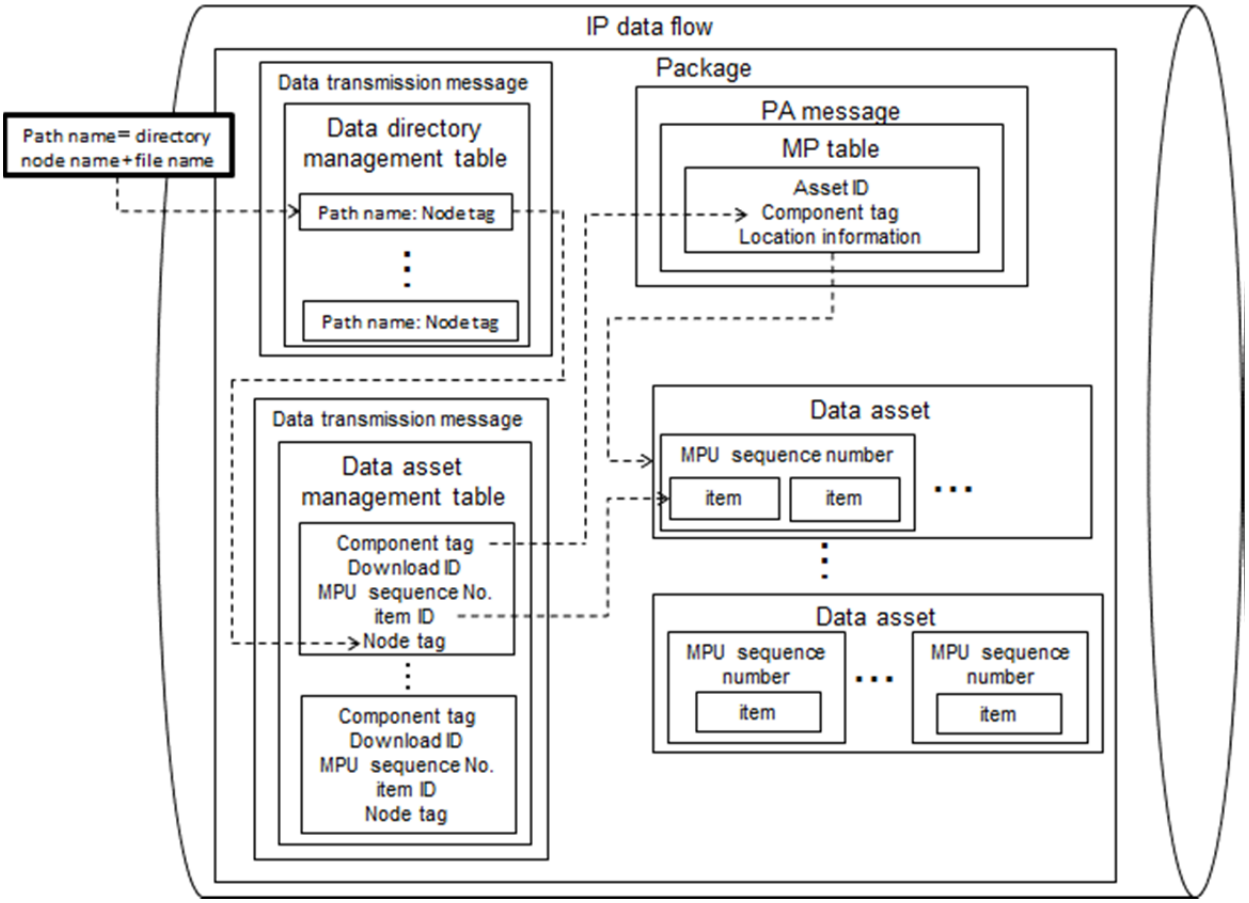


Fig. 10-1 Overview of acquiring file which configures application

For the file that configures application, the path name is specified in the description of the application such as HTML5. Here, the path name shall be described by a set of the directory node name and the file name. Also, the node tag is specified as the identifier which integrates the directory node and the file, and is applied as the information to link each table. The receiver whose path name is specified by application acquires the node tag of file with the specified path name from the data directory management table which is transmitted by the data transmission

message. Next, from the data asset management table which is transmitted by another data transmission message, the receiver acquires the component tag, the download ID, the MPU sequence number and the item ID in the asset to which item with acquired the node tag is transmitted. By referring to the MP table, the receiver acquires the location information of asset which has acquired component tag, and it specifies the asset whose file is actually transmitted. In the specified asset, by the acquired download ID and the download ID which is recorded in multi-extension header region of the MMTP packet which transmits item, the unit of repeated transmission of the file corresponding to the caroucel can be uniquely identified. And among items which are repeatedly transmitted, the file that acquired the MPU sequence number and the item ID is the desired file.

Moreover, by always monitoring the data asset management table, the receiver can detect that the file configuring application has been updated, and can always show application in the latest state. The detection of update becomes possible on each stage; the unit of asset, the unit of MPU and the unit of file. By the data asset management table, the component tag of updated file, the MPU sequence number and the item ID are acquired. As with the file acquisition mentioned above, referring to the MP table, the asset is specified, and it follows that obtained MPU sequence number and item with the item ID are the desired updated file.

As the data content management table realizes a flexible and effective cache control, it supplies the configuration information of file as a data content. It defines the unit which is presented in the data content (ex. page) as the presentation unit (PU), and by the unit, it shows the information about which items are related and which presentation unit is linkaged to, and so on. By these information, the other item that is involved in the same presentation unit as the first required item can be known. Also, the information of the cache control by the unit of PU, etc. can be acquired, too. As the result, the receiver can realize to shorten the acquisition time of the whole page and to cache the next page preferentially

10.2 Application Transmission System

10.2.1 Configuration of MPU and Storing into MMTP Payload

As for the application transmission, the data which configures the application is capsulized into asynchronous type MPU (MPU with non-timed media data) by the unit of file, and transmitted. The MPU is composed of the MPU metadata and more than or equal to one MFU. The MPU metadata is composed of ftyp box, mmpu box, moov box, and meta box. The file body is stored in one idat box in the meta box, and the file body shall be the MFU.

When transmitting, the MPU metadata is not transmitted, but only MFU is transmitted. In case that the size of MFU is small, plural MFU's which compose the MPU are aggregated and transmitted as one MMTP payload. In the case that the size of MFU's is big and is not transmitted in one IP packet, it is transmitted as plural MMTP payloads by dividing the MFU's. Sequence number of MPU is recorded in the MMTP payload. This value shall be unique at least in the asset which is transmitted. An example that the MPU is composed of the file and is stored in the MMTP payload is shown in Fig. 10-2.

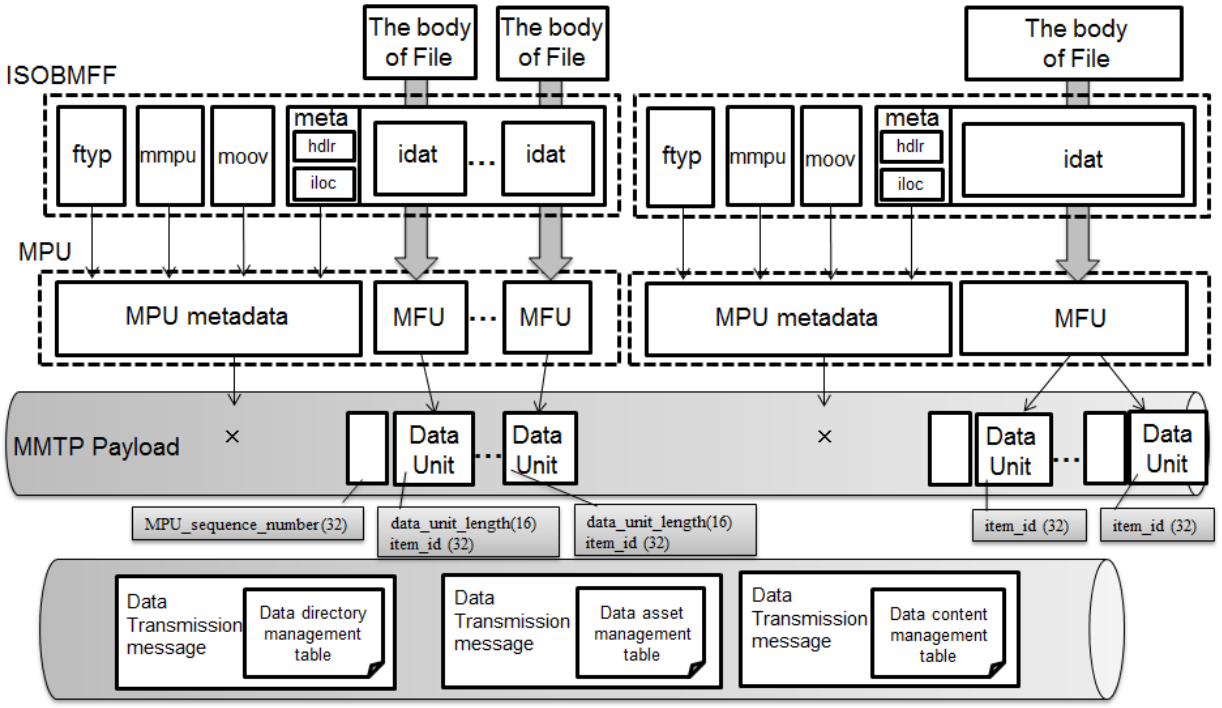


Fig. 10-2 An example of storing application in MMTP Payload

10.2.2 Configuration of MFU of Application Transmission

The MFU shall be configured as shown in Table 10-1.

Table 10-1 Configuration of MFU

Data structure	Number of bit	Data notation

MFU_data_byte () {
for(i=0; i<N; i++) {
data
}
}
	8	uimsbf

The meaning of MFU:

data (data): The body of item which is not compressed or compressed is stored.

10.2.3 Transmission of Application File by Fragmentation

In the case of the transmitting file which configures the application by dividing into more than 255, in each MMTP packet which transmits divided the file data, the multi-type header extension shown in 10.2.4 is arranged in the region of MMTP extension header. Even in the case of the transmitting file which configures the application by dividing into less than 256, this multi-type header extension may be arranged. By this, in spite of the number of the divided file, it is possible to easily reconfigure the file when receiving. Especially in the case of receiving from the middle of the file, or in the case of failing in receiving each divided fragment, it is effective as the control information for reconfiguring the file by receiving repeatedly or acquiring communication.

10.2.4 Multi-type Header Extension Related to Transmission of Application

In the transmission of application, the following multi-type header extension is arranged as the need arises.

10.2.4.1 Multi-type Header Extension Involving Download Identification Information

The multi-type header extension involving the download identification information, as shown in Table 10-2, is arranged in the MMTP packet which transmits the application data in corresponding to the download identification which is specified in the data asset management table.

Table 10-2 Configuration of multi-type header extension involving download identification information

Data structure	Number of bit	Data notation
Header_extension_byte_for_download_id () {		
hdr_ext_end_flag	1	bslbf
hdr_ext_type	15	uimsbf
hdr_ext_length	16	uimsbf
download_id	32	uimsbf
}		

The meaning of multi-type header extension involving download identification information:

hdr_ext_end_flag (multi-type header extension end flag): In the case that the multi-type header extension immediately after is the last of header extension, it shall be '1', and otherwise it shall be '0'.

hdr_ext_type (multi extension header type): This represents the extension type of the multi-type header extension. 0x0002 is described.

hdr_ext_length (multi extension header length): From immediately after this field, the size of one extension header region immediately after (the size of hdr_ext_byte immediately after) is represented by the unit of byte. For the multi-type header extension involving the download identification information, it is 0x0004.

download_id (download identification): This plays a role of identifying the data content uniquely. It is connected to the download identification which is arranged in data asset management table.

10.2.4.2 Multi-type Header Extension Involving File Fragmentation Transmission Information

When the file shown in 10.2.3 is transmitted by divding, the multi-type header extension involving the file fragmentation transmission information shown in Table 10-3 is arranged in the MMTP packet which transmits the application data

Table 10-3 Configuration of multi-type header extension involving file fragmentation transmission information

Data structure	Number of bit	Data notation
Header_extension_byte_for_item_fragmentation () {		
hdr_ext_end_flag	1	bslbf
hdr_ext_type	15	uimsbf
hdr_ext_length	16	uimsbf
item_fragment_number	32	uimsbf
last_item_fragment_number	32	uimsbf
}		

The meaning of multi-type header extension involving file fragmentation transmission information:

hdr_ext_end_flag (multi-type header extension end flag): In the case that the multi-type header extension immediately after is the last of header extension, it shall be '1', and otherwise it shall be '0'.

hdr_ext_type (multi extension header type): This represents the extension type of the multi-type header extension. 0x0003 is described.

hdr_ext_length (multi extension header length): From immediately after this field, the size of one extension header region immediately after (the size of hdr_ext_byte immediately after) is represented by the unit of byte. For the multi-type header extension involving the file fragmentation transmission information, it is 0x0008.

item_fragment_number (item fragment number): This represents the number of the concerned MMTP payload as one fragment which was divided in the whole item (file) that is represented by the same item_id. The number shall be the value which increases one by one from zero.

last_item_fragment_number (last item fragment number): This represents the last item fragment number of divided fragment in the whole item (file) which is represented by the same item_id. This becomes the value that the whole number of the item fragment (MMTP payload) minus one.

10.2.5 Index Item

The index item is provided by collecting the information of the item (file) involved in the MPU, and it is placed as one item which composes the MPU. In the case that the index item flag is one in the DAM table mentioned in the following, the index item shown in Table 10-4 is transmitted as the MFU of the component of corresponding MPU.

Table 10-4 Configuration of Index Item

Data structure	Number of bit	Data notation
Index_item () {		
num_of_items	16	uimsbf
for (i=0; i<num_of_items; i++) {		
item_id	32	uimsbf
item_size	32	uimsbf
item_version	8	uimsbf
file_name_length	8	uimsbf
for (j=0; j<file_name_length; j++) {		
file_name_byte	8	char
}		
checksum_flag	1	bslbf
reserved_future_use	7	bslbf
if (checksum_flag == 1) {		
item_checksum	32	uimsbf
}		
item_type_length	8	uimsbf
for (j=0; j<item_type_length; j++) {		
item_type_byte	8	char
}		
compression_type	8	uimsbf
if (compression_type != 0xFF) {		
original_size	32	uimsbf
}		
}		
}		

The meaning of index item:

num_of_items (number of items): This represents the number of items except the index item involved in the MPU.

item_id (item identification): This represents ID which identifies the item.

item_size (item length): This represents the byte length of the item.

item_version (item version): This represents the version of the item.

file_name_length (file name length): This represents the byte length of the following file name region.

file_name_byte (file name byte): The file name of the concerned item is stored in a series of region.

checksum_flag (checksum flag): This represents whether checksum is recorded or not.

item_checksum (item checksum): This represents CRC32 which is calculated by the MFU which the body of item is stored in as checksum. CRC shall follow the ITU-T Recommendation H.220.

item_type_length (item type length): This represents the byte length of the following item

type region.

item_type_byte (item type byte): This represents the media type of the concerned item in a series of region, based on RFC1521, RFC1590.

compression_type (compression identification): This field of 8 bits specifies the compression format which is used for compressing the item. The value which identifies the compression format is provided in operation. But 0xFF shall represent non-compression.

original_size (original size): This field of 32 bits represents the size of the item before compression by the number of byte.

10.3 Control Information for Application Transmission System

10.3.1 Message Used for Application Transmission System

10.3.1.1 Data Transmission Message

The data transmission message is the message in which the table related to the data transmission is stored. Just like the M2 section message, all of the tables to be stored shall be the section extension format. The configuration of the data transmission message is shown in Table 10-5.

Table 10-5 Configuration of data transmission message

Data structure	Number of bit	Data notation
Data_Transmission_Message () {		
message_id	16	uimsbf
version	8	uimsbf
length	32	uimsbf
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'1'	1	bslbf
'11'	2	bslbf
section_length	12	uimsbf
data_transmission_session_id	8	uimsbf
reserved_future_use	8	bslbf
'11'	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for(i=0; i<N; i++) {		
signaling_data_byte	8	bslbf
}		
CRC_32	32	rpchof
}		

The meaning of data transmission message:

table_id (table identification): This represents the identification information of table which is

stored in data transmission message.

section_syntax_indicator (section syntax indicator): This shall be '1' which represents the extension format.

section_length (section length): This represents the number of data byte which follows from immediately after the section length region.

data_transmission_session_id (data transmission session identification): This represents the common session identification information between tables which are transmitted by the data transmission message. The same value is set for the tables which have a reference relationship. In the case that the content of any table has changed, each table is updated synchronously so that the reference relationship is maintained.

version_number (version number): This represents the version number of the table.

current_next_indicator (current next indicato): In case that the table can be used now, this shall be '1', and in the case of representing the table cannot be used now and will be effective next, it shall be '0'.

section_number (section number): This represents the section number configuring the table.

last_section_number (last section number): This represents the last section number configuring the table.

CRC_32 (CRC): This shall follow ITU-T Recommendation H.220. The calculation range shall be from the table identification field to immediately before this field.

10.3.2 Table Used for Application Transmission System

10.3.2.1 MH-Application Information Table (MH-AIT)

All information about the application and startup state required by the application, etc. are stored in MH-AIT. By using the data in MH-AIT, it is possible for the broadcaster to indicate the receiver to change startup state to the application. MH-AIT is transmitted by storing in the M2 section message. The configuration of MH-AIT is shown in Table 10-6.

Table 10-6 Configuration of MH-application information table

Data structure	Number of bit	Data notation
MH-Application_Information_Table () {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
application_type	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
reserved_future_use	4	bslbf
common_descriptor_length	12	uimsbf
for (i=0; i<N; i++) {		
descriptor ()		
}		
reserved_future_use	4	bslbf
application_loop_length	12	uimsbf
for (i=0 ;i<N; i++) {		
application_identifier ()		
application_control_code	8	uimsbf
reserved_future_use	4	bslbf
application_descriptor_loop_length	12	uimsbf
for (j=0; j<M; j++) {		
descriptor ()		
}		
}		
CRC_32	32	rpchof
}		

The meaning of MH-application information table:

section_syntax_indicator (section syntax indicator): Section syntax indicator is a field of one bit, and it shall always be '1'.

section_length (section length): This represents the number of byte of the section from immediately after section length field to the last of section including CRC32. In order that the length of all sections does not exceed 4096, the section length shall not exceed 4093 (0xFFD by hexadecimal).

application_type (application format): This represents the type of the application which is object to control by MH-AIT. The assignment of the application type is shown in Table 10-7.

Table 10-7 Application type

Application type	description
0x0000	reserved_future_use
0x0001	ARIB-J application
0x0002 – 0x000F	reserved_future_use
0x0010	Liaison between broadcasting and communication HTML 5 application
0x0011	ARIB-HTML 5 application
0x0012 – 0x7FFF	reserved_future_use

version_number (version number): This field of 5 bits represents the version number of sub-table. One is added to the version number when the information in the sub-table has been changed. When the value becomes 31, it will return to 0 next.

current_next_indicator (current next indicator): The indication by this one bit shall always be '1'.

section_number (section number): This field of 8 bits represents the section number. The section number of the first section in the sub-table is 0x00. In each time the section which has the same table identification and application format is added, one is added to the section number.

last_section_number (last section number): This field of 8 bits specifies the last section number in the sub-table which the section belongs to.

common_descriptors_length (common descriptor loop length): This field of 12 bits specifies the byte length of the following common descriptor region. The descriptor in this descriptor region is applied to all applications in MH-AIT sub-table.

application_loop_length (application information loop length): This field of 12 bits specifies the byte length of the whole loop in which the following application information is stored.

application_identifier () (application identifier): The value which application is uniquely identified. This value is specified for each application format.

application_control_code (application control code): This field of 8 bits specifies the control code to control the state of the application. The semantics of this field is specified for each application format. If not specified for each application format, it shall be the semantics shown in Table 10-8.

Table10-8 Application control code

Value of application control code	Identification name	meaning
0x01	AUTOSTART	To startup the application
0x02	PRESENT	This shows that the state of the application is possible to be executed.
0x04	KILL	To shut down the application
0x05	PREFETCH	To acquire the application and hold it

application_descriptors_loop_length (application information descriptor loop length): This field of 12 bits specifies the byte length of the following descriptor region. The descriptor in this descriptor region is applied only to specified the application.

CRC_32 (CRC): This shall follow the ITU-T Recommendation H.222.0.

10.3.2.2 Data Directory Management Table (DDM Table)

The data directory management table supplies the directory configuration of the file configuring the application, in order to separate the file configuration of the application and the configuration for transmitting the file. One table of this is stored in the data transmission message. The configuration of the data directory management table is shown in Table 10-9.

Table 10-9 Configuration of data directory management table

Data structure	Number of bit	Data notation
Data_Directory_Management_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
data_transmission_session_id	8	uimsbf
reserved_future_use	8	bslbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
base_directory_path_length	8	uimsbf
for (j=0; j<base_directory_path_length; j++) {		
base_directory_path_byte	8	uimsbf
}		
num_of_directory_nodes	8	uimsbf
for (j=0; j<num_of_directory_nodes; j++) {		
node_tag	16	uimsbf
directory_node_version	8	uimsbf
directory_node_path_length	8	uimsbf
for (k=0; k<directory_node_path_length; k++) {		
directory_node_path_byte	8	char
}		
num_of_files	16	uimsbf
for (k=0; k<num_of_files; k++) {		
node_tag	16	uimsbf
file_name_length	8	uimsbf
for (m=0; m<file_name_length; m++) {		
file_name_byte	8	char
}		
}		
}		
CRC_32	32	rpchof
}		

The meaning of data directory management table:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and it shall be always '1'.

section_length (section length): This represents the number of bytes of the section from immediately after the section length field to the last of the section involving CRC32. In order that the the length of all sections does not exceed 4096, the section length shall not exceed 4093 (0xFFD by hexadecimal).

data_transmission_session_id (data transmission session identification): This represents the session identification information common to tables which are transmitted by the data transmission message.

version_number (version number): This field of 5 bits is the version number of the sub-table. In the case that the information in the sub-table is changed, one is added to the version number. When the value becomes 31, it will return to 0 next.

current_next_indicator (current next indicator): The indication of this one bit shall be always '1'.

section_number (section number): This field of 8 bits represents the section number. The section number of the first section in the sub-table is 0x00. One is added to the section number in each time the section which has the same table identification and application format is added.

last_section_number (last section number): This field of 8 bits specifies the last section number in the sub-table which the section belongs to.

base_directory_path_length (base directory path length): This represents the byte length of the following base directory path region.

base_directory_path_byte (base directory path byte): The base directory path is stored in a series of region.

num_of_directory_nodes (number of directory nodes): This represents the number of the directory nodes which are recorded in this section of the data directory management table.

node_tag (node tag): This represents a label to identify directory or file, as a directory or a node tag of the file.

directory_node_version (directory node version): This represents the version of the directory node.

directory_node_path_length (directory node path length): This represents the byte length of the following directory node path region.

directory_node_path_byte (directory node path byte): The directory node path is stored in a series of region.

num_of_files (number of files): This represents the number of files which are recorded in the data directory management table.

file_name_length (file name length): This represents the byte length of the following file name region.

file_name_byte (file name byte): The concerned file name is stored in a series of region.

10.3.2.3 Data Asset Management Table (DAM Table)

The data asset management table supplies the configuration of MPU in the asset and the

version information of each MPU. It correspnds to DII of DSM-CC defined in the ISO/IEC 13818-6, and by watching the version of MPU, a means to be enable to update the information by the unit of MPU is supplied. One table of this is stored in the data transmission message. The configuration of the data asset management table is shown in Table 10-10.

Table 10-10 Configuration of data asset management table

Data structure	Number of bit	Data notation
Data_Asset_Management_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
data_transmission_session_id	8	uimsbf
reserved_future_use	8	bslbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
transaction_id	32	uimsbf
component_tag	16	uimsbf
download_id	32	uimsbf
num_of_mpus	8	uimsbf
for (j=0; j<num_of_mpus; j++) {		
mpu_sequence_number	32	uimsbf
mpu_size	32	uimsbf
index_item_flag	1	bslbf
index_item_id_flag	1	bslbf
index_item_compression_type	2	uimsbf
reserved_future_use	4	bslbf
if(index_item_flag==1 && index_item_id_flag==1){		
index_item_id	32	uimsbf
}		
num_of_items	16	uimsbf
for (k=0; k<num_of_items; k++) {		
node_tag	16	uimsbf
if(index_item_flag==0) {		
item_id	32	uimsbf
item_size	32	uimsbf
item_version	8	uimsbf
checksum_flag	1	bslbf
reserved_future_use	7	bslbf
if (checksum_flag == 1) {		
item_checksum	32	uimsbf

}		
item_info_length	8	uimsbf
for (l=0; l<item_info_length; l++) {		
item_info_byte	8	uimsbf
}		
}		
}		
mpu_info_length	8	uimsbf
for (m=0; m<mpu_info_length; m++) {		
mpu_info_byte	8	uimsbf
}		
}		
component_info_length	8	uimsbf
for (n=0; n<component_info_length; n++) {		
component_info_byte	8	uimsbf
}		
CRC_32	32	rpchof
}		

The meaning of data asset management table:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and it shall be always '1'.

section_length (section length): This represents the number of byte of the section from immediately after the section length field to the last of the section involving CRC32. In order that the length of all sections does not exceed 4096, the section length shall not exceed 4093 (0xFFD by hexadecimal).

data_transmission_session_id (data transmission session identification): This represents the session identification information common to the tables which are transmitted by the data transmission message.

version_number (version number): This field of 5 bits is the version number of the sub-table. In the case that the information in the sub-table is changed, one is added to the version number. When the value becomes 31, it will return to 0 next.

current_next_indicator (current next indicator): The indication of this one bit shall be always '1'.

section_number (section number): This field of 8 bits represents the section number. The section number of the first section in the sub-table is 0x00. One is added to the section number in each time the the section which has the same table identification and application format is added.

last_section_number (last section number): This field of 8 bits specifies the last section number in the sub-table which the section belongs to.

transaction_id (transaction identification): This is an identifier which has a version function of the data component.

component_tag (component tag): This represents a label to identify the component.

download_id (download identification): This plays a role of label to uniquely identify the data content. For the MMTP packet which transmits the application, the multi-extension header type

0x0002 is specified as needed, and the download identification is recorded in the multi-extension header region. When thr data event is operated by the provision of the coding system and so on, the data_event_id is coded into bit 28-31 of the download identification. In other case, the range and the value which should be guaranteed on uniqueness are provided in the opration.

num_of_mpus (number of MPUs): This represents the number of MPU's which are described in this section.

mpu_sequence_number (MPU sequence number): This represents the sequence number of MPU.

mpu_size (MPU size): This represents the sum of the number of bytes for all items involved in MPU.

index_item_flag (index item flag): When the index item is involved in MPU, this represents '1'.

index_item_id_flag (index item identification flag): In the case of specifying the item identification of the indx item, this represents '1'. In the case of fixing the value of the item identification in the operation, it represents '0'.

index_item_compression_type (index item compression type): This represents the compression format of the index item. The value to identify the compression format is provided in the operation. But '11' shall represent non-compression.

index_item_id (index item identification): This represents the item identification of the index item.

num_of_items (nuber of items): This represents the number of items which configure MPU.

node_tag (node tag): This represents a label to identify the item as a node tag corresponding to the item.

item_id (item identification): This represents ID to identify the item.

item_size (item size): This represents the byte length of the item.

item_version (item version): This represents the version of the item.

checksum_flag (checksum flag): This represents whether there is a record of checksum or not.

item_checksum (item checksum): This represents CRC32 which is calculated by MFU which the body of the item is stored in as checksum. CRC shall follow the ITU-T Recommendation H.220.

item_info_length (item information length): This represents the byte length of the following item information region.

item_info_byte (item information byte): The information about the concerned item is stored as one or plural descriptors in a series of regions.

mpu_info_length (MPU information length): This represents the byte length of the following MPU information region.

mpu_info_byte (MPU information byte): The information about the concerned MPU is stored as one or plural descriptors in a series of regions.

component_info_length (component information length): This represents the byte length of the following component information region.

component_info_byte (component information byte): The information about the concerned component is stored as one or plural descriptors in a series of regions.

10.3.2.4 Data Content Configuration Table (DCC Table)

In order to realize a flexible and effective cache control, the data content configuration table supplies the configuration information of the file as the data content. It shows which file or directory is related with, which presentation unit is linked with, etc. by the unit of the presentation unit (PU) in the data content. One table of this is stored in the data transmission message. The configuration of the data content configuration table is shown in Table 10-11.

Table 10-11 Configuration of data content configuration table

Data structure	Number of bit	Data notation
Data_Content_Configuration_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
data_transmission_session_id	8	uimsbf
reserved_future_use	8	bslbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
content_id	16	uimsbf
content_version	8	uimsbf
content_size	32	uimsbf
PU_info_flag	1	bslbf
content_info_flag	1	bslbf
reserved_future_use	6	bslbf
if (PU_info_flag == 1) {		
number_of_PUs	8	uimsbf
for (j=0; j<number_of_PUs; j++) {		
PU_tag	8	uimsbf
PU_size	32	uimsbf
number_of_member_nodes	8	uimsbf
for (k=0; k<number_of_member_nodes; k++) {		
node_tag	16	uimsbf
}		
PU_descriptor_loop_length	8	uimsbf
for (k=0; k<PU_descriptor_loop_length; k++) {		
PU_descriptors_byte	8	uimsbf
}		
}		
}		
else {		
number_of_nodes	16	uimsbf

for (l=0; l<number_of_nodes; l++) {		
node_tag	16	uimsbf
}		
}		
if (content_info_flag == 1) {		
content_descriptor_loop_length	8	uimsbf
for (j=0; j<content_descriptor_loop_length; j++) {		
content_descriptors_byte	8	uimsbf
}		
}		
CRC_32	32	rpchof
}		

The meaning of data content configuration table:

section_syntax_indicator (section syntax indicator): The section syntax indicator is a field of one bit, and it shall be always '1'.

section_length (section length): This represents the number of bytes of the section from immediately after the section length field to the last of the section involving CRC32. In order that the length of all sections does not exceed 4096, the section length shall not exceed 4093 (0xFFD by hexadecimal).

data_transmission_session_id (data transmission session identification): This represents the session identification information common to tables which are transmitted by the data transmission message.

version_number (version number): This field of 5 bits is the version number of the sub-table. In the case that the information in the sub-table is changed, one is added to the version number. When the value becomes 31, it will return to 0 next.

current_next_indicator (current next indicator): The indication of this one bit shall be always '1'.

section_number (section number): This field of 8 bits represents the section number. The section number of the first section in the sub-table is 0x00. One is added to the section number in each time the section which has the same table identification and application format is added.

last_section_number (last section number): This field of 8 bits specifies the last section number in the sub-table which the section belongs to.

content_id (content identification): This plays a role of label to uniquely identify the data content.

content_version (content version): This shall be a region in which the version number of the data content is recorded.

content_size (content size): This shall be a region in which the size of the data content is recorded.

PU_info_flag (presentation unit information flag): This represents whether the concerned table is the information of the presentation unit or not.

content_info_flag (content information flag): This represents whether there is a region of the content descriptor or not.

number_of_PUs (number of presentation units): This shall be a region in which the number of presentation units is recorded.

PU_tag (presentation unit tag): This represents a label to identify the presentation unit.

PU_size (presentation unit size): This shall be a region to write the size of the presentstion unit.

number_of_member_nodes (number of member nodes): This represents the number of files configuring PU or node designations of the directory.

node_tag (node tag): This represents the file configuring PU or the node tag of the directory.

PU_descriptor_loop_length (PU descriptor region length): This represents the number of whole bytes of the following PU descriptor region.

PU_descriptors_byte (PU descriptor data byte): Data of the descriptor about PU is stored in a series of region.

number_of_nodes (number of nodes): This represents the number of files configuring the content or the node designations of the directory.

content_descriptor_loop_length (content descriptor region length): This represents the number of whole bytes of the following content descriptor region.

content_descriptors_byte (content descriptor data byte): The data of descriptor about the data content are stored in a series of regions.

10.3.3 Descriptor Used in MH-Application InformationTable

10.3.3.1 MH-Application Descriptor

In the application information descriptor loop of MH-AIT, one MH-application descriptor is always arranged for each application. The configuration of the MH-application descriptor is shown in Table 10-12.

Table 10-12 Configuration of MH-application descriptor

Data structure	Number of bit	Data notation
MH-Application_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
application_profiles_length	8	uimsbf
for(i=0; i<N; i++) {		
application_profile	16	uimsbf
version.major	8	uimsbf
version.minor	8	uimsbf
version.micro	8	uimsbf
}		
service_bound_flag	1	bslbf
visibility	2	bslbf
reserved_future_use	4	bslbf
present_application_priority	1	bslbf
application_priority	8	uimsbf
for(i=0; i<N; i++) {		
transport_protocol_label	8	uimsbf
}		
}		

The meaning of MH-application descriptor:

application_profiles_length (application profile information length): This represents the byte length of the whole application profile information involved in the following loop.

application_profile (application profile): This represents the application profile of the receiver that this application can be executed. If the receiver implements this profile, it shows that it has a faculty to execute this application. The content of the profile is defined by each application format, respectively.

version.major (major version): This represents the major version of the above-mentioned profile.

version.minor (minor version): This represents the minor version of the above-mentioned profile.

version.micro (micro version): This represents the micro version of the above-mentioned profile.

Minimum profile to execute this application is represented by the above-mentioned four fields. In the case that a profile fits at least one logical operation shown in the following and there is a profile which results in true in this application profile information, the receiver shall startup this application.

(profile of application ∈ a set of profile which is implemented in the terminal)

AND { (major version of application < major version of the terminal for this profile)

OR [(major version of application = major version of the terminal for this profile))

AND ({ minor version of application < minor version of the terminal for this profile)}

OR { [minor version of application ＝ minor version of the terminal for this profile)]

AND [micro version of application ≦ micro version of the terminal for this profile)]})] }

service_bound_flag (service boundary flag): This represents whether this application is valid only for present service or not. In the case that this flag indicates 1, the application is connected to only present service, and in the case of changing to the other service, the application will finish.

visibility (visibility): This represents whether visible or not for users and the other application in executing this application. Refer to Table 10-13.

Table 10-13 Visibility

Value of visibility	meaning
00	This application shall be invisible except reporting error such as log output.
01	This application shall be invisible for user, but visible for the other application through API and so on.
10	reserved_future_use
11	This application shall be visible for either user or the other application.

present_application_priority (present application priority): When the application is continuing to work with designated as PRESENT, this represents the priority between this application and another application designated as AUTOSTART. In the case this value is '0', this application is finished and the application designated as AUTOSTART starts up. In the case this value is '1', this application continues working. In order to control continuous working before and after updating MH-AIT of the application, this is referred to only when updating MH-AIT.

application_priority (application priority): This represents the relative priority among the applications in the case that multiple applications work.

transport_protocol_label (transport protocol label): This represents the value that uniquely identifies the transport protocol which is transmitting the application. It corresponds to the field with the same name as that of the MH-transport protocol descriptor.

10.3.3.2 MH-Transport Protocol Descriptor

As a means of transmitting the application, aiming to designate the transport protocol for broadcasting and communication, etc. and to show the location information of the application which depends upon the transport protocol, in the common descriptor loop of MH-AIT or the application information descriptor loop, the number of the transport protocol label of the MH-application descriptor are arranged. The configuration of the MH-transport protocol descriptor is shown in Table 10-14.

Table 10-14 Configuration of MH-transport protocol descriptor

Data structure	Number of bit	Data notation
MH-Transport_Protocol_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
protocol_id	16	uimsbf
transport_protocol_label	8	uimsbf
for(i=0; i<N; i++) {		
selector_byte	8	uimsbf
}		
}		

The meaning of MH-transport protocol descriptor:

protocol_id (protocol identification): This represents the protocol which transmits the application. Refer to Table 1-15.

Table 10-15 Protocol identification

value	meaning
0x0000	reserved_future_use
0x0001 – 0x0002	reserved
0x0003	HTTP/HTTPS transmission
0x0004	Data carousel transmission
0x0005	MMT non-timed transmission
0x0006 – 0xFFFF	reserved_future_use

transport_protocol_label (transport protocol label): This represents the value which uniquely identifies the transmission means of the application. It corresponds to the field with the same name as that of the MH-application descriptor.

selector_byte (selector region): The supplementary information which is provided for each protocol identification is stored here. The data structures in the cases of the HTTP/HTTPS transmission and the MMT non-timed transmission are shown in Table 10-16.

Table 10-16 Selector byte in the cases of the HTTP/HTTPS transmission and the MMT non-timed transmission

Data structure	Number of bit	Data notation
for (i=0; i<N; i++) {		
URL_base_length	8	uimsbf
for(j=0; j<URL_base_length; j++) {		
URL_base_byte	8	uimsbf
}		
URL_extension_count	8	uimsbf
for(j=0; j<URL_extension_count; j++) {		
URL_extension_length	8	uimsbf
for(k=0; k<URL_extension_length; k++) {		
URL_extension_byte	8	uimsbf
}		
}		
}		

The meaning of selector byte:

URL_base_length (URL base length): This represents the number of bytes in the URL base to acquire the application.

URL_base_byte (URL base): The letter sequence in the URL base to acquire the application is stored here.

URL_extension_count (URL extension count): This represents the number of the extension part in URL to acquire the application. In the case that it is plural, it represents plural locations that can acquire the application exist in the same URL base region.

URL_extension_length (URL extension length): This represents the number of bytes of the extension part in the URL base to acquire the application.

URL_extension_byte (URL extension): This is the letter sequence in the extension part of URL to acquire the application. Also, as the whole is loop, it represents that plural base regions in URL to acquire the application can be set.

10.3.3.3 MH-Simple Application Location Descriptor

The MH-simple application location descriptor is always arranged one by one for each application in the application information descriptor loop of MH-AIT, aiming to designate details about acquisition of the application. The configuration of the MH-simple application location descriptor is shown in Table 10-17.

Table 10-17 Configuration of MH-simple application location descriptor

Data structure	Number of bit	Data notation
MH-Simple_Application_Location_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for(i=0; i<N; i++) {		
initial_path_byte	8	uimsbf
}		
}		

The meaning of MH-simple application location descriptor:

initial_path_byte (application URL): A character string representing the URL of the entry point of the corresponding application. Considering the acquirable location of the application shown by the MH-transport protocol descriptor as a root, it is represented by a relative path.

10.3.3.4 MH-Application Boundary and Permission Descriptor

Aiming to set the application boundary and to set the permission of broadcasting resource access for each region (URL), one or plural MH-application boundaries and permission descriptors are arranged in the application information descriptor loop of MH-AIT. In the case that this descriptor is not arranged, the application boundary becomes infinite, and all accesses to broadcasting resources shall be permitted. The configuration of the MH-application boundary and the permission descriptor is shown in Table 10-18.

Table 10-18 Configuration of MH-application boundary and permission descriptor

Data structure	Number of bit	Data notation
MH-Application_Boundary_and_Permission_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for(i=0; i<N; i++){		
permission_bitmap_count	8	uimsbf
for(j=0; j<permission_bitmap_count; j++){		
permission_bitmap	16	bslbf
}		
managed_URL_count	8	uimsbf
for(j=0; j<managed_URL_count; j++){		
managed_URL_length	8	uimsbf
for(k=0; k<managed_URL_length; k++){		
managed_URL_byte	8	uimsbf
}		
}		
}		
}		

The meaning of MH-application boundary and permission descriptor:

permission_bitmap_count (number of access permission bitmap): This represents the number of permission_bitmap to be designated.

permission_bitmap (access permission bitmap): It is configured by a bitmap for each function whether access to each broadcasting resource is permitted or not. Upper three bits represents the change of the bitmap. Assignment of the functional bitmap is provided in operation.

managed_URL_count (number of access permission management region setting): This represents the number of the region in which the setting of the access permission represented by permission_bitmap is applied. Also, if this value indicates 0, it shall represent all regions. In other word, it is interpreted that it means the URL involving an arbitrary location. But, in the case that a specific region is designated as the setting of the other access permission bitmap, the setting is valid (based on the rule that the setting of the access permission for narrow region is dominant).

managed_URL_length (byte length of access permission management region setting): This represents the number of bytes for the setting of the access permission management region (character string of URL).

managed_URL_byte (information of access permission management region setting): This represents a character string of the URL of the access permission management region. It specifies the domain or its sub-directory.

10.3.3.5 MH-Autostart Priority Descriptor

At the maximum one MH-autostart priority descriptor is arranged for the application in the application information descriptor loop of MH-AIT, aiming to specify the application startup priority. But this descriptor shall be arranged only to describe the application information that the application control code represents those which specify auto startup of the application (AUTOSTART, etc.). When this descriptor is not arranged, it may be considered as the priority is lowest, involving the data broadcasting. The configuration of the MH-autostart priority descriptor is shown in Table 10-19.

Table 10-19 Configuration of MH-autostart priority descriptor

Data structure	Number of bit	Data notation
MH-Autostart_Priority_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
autostart_priority	8	uimsbf
}		

The meaning of MH-autostart priority descriptor:

autostart_priority (autostart priority): This represents the autostart priority of corresponding applications among the data broadcasting which is linked to the receiving service now and all applications.

10.3.3.6 MH-Cache Control Information Descriptor

The MH-cache control information descriptor is arranged at most one for each application in the application information descriptor loop of MH-AIT, aiming to be used as the cache control in the case of caching and holding the resource configuring the application, when the application is supposed to be reused.

In the case that this descriptor is not arranged, the receiver does not hold the resource configuring the application but delete it when the application ends. The configuration of the MH-cache control information descriptor is shown in Table 10-20.

Table 10-20 Configuration of MH-cache control information descriptor

Data structure	Number of bit	Data notation
MH-Cache_Control_Info_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
application_size	16	uimsbf
cache_priority	8	uimsbf
package_flag	1	bslbf
application_version	7	uimsbf
expire_date	16	bslbf
}		

The meaning of MH-cache control information descriptor

application_size (application size): This represents the size of whole application by the unit of kilo byte. In the case that the size is not clear, 0 is designated.

cache_priority (cache priority): This represents priority to hold the cache of application. It shall be considered that the bigger the value is, the higher the priority is. It is supposed that in the case of exceeding a cache capacity, the cache is deleted by the unit of the application considering this information as hint information. When the priority is not designated, 0xFF is designated.

package_flag (package flag): This represents whether the application is packaged and collected up to one file or not. If it is 1, it represents packaged.

application_version (application version): This represents the version number of the application. The receiver memorizes the application version corresponding to the application which is cached. After that, if the application version is updated when the application starts up, it does not use the application that is held in the cache, but it newly acquires and uses the application from the designated URL, and also rewrites the content of the cache.

expire_date (period of validity of cache): The validity period of the cache for the application is represented by the lower 16 bits of MJD as year, month and day. The receiver may cache until this date. If passing this date, the application is deleted from the cache. In the case of no expiration date, 0xFFFF is designated.

10.3.3.7 MH-Randomized Latency Descriptor

The MH-randomized latency descriptor is arranged at most one for each application in the application information descriptor loop of MH-AIT, aiming to delay the timing of application control by a delay amount which is set probablistically, supposing to disperse a load of server

accesses for the application acquisition. When this descriptor is not arranged, the application shall be controlled by the control code at the timing when the MH-AIT of a specific version is received first. The configuration of the MH-randomized latenct descriptor is shown in Table 10-21.

Table 10-21 Configuration of MH-randomized latency descriptor

Data structure	Number of bit	Data notation
MH-Randomized_Latency_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
range	16	uimsbf
rate	8	uimsbf
randomization_end_time_flag	1	bslbf
reserved_future_use	7	bslbf
if(randomization_end_time_flag == 1){		
randomization_end_time	40	uimsbf
}		
}		

The meaning of MH-randomized latency descriptor:

range (range of delay time): This represents the maximum delay time from the present time until the control code is applied. It is designated by the number of second.

rate (dispersion number): This represents the number of steps of the delay time to the application of the control code which is set probablistically. The receiver calculates the delay time Td by an algebraic equation Td=N×range÷rate, based on the value N which is randomly selected among integer values from 0 to the rate, and it applies the control code by delaying Td from the reception time of MH-AIT.

randomization_end_time_flag (randomization end time flag): This represents whether randomization end time (randomization_end_time) is designated or not. In the case of 1, it shall be designated.

randomization_end_time (randomization end time): This is the period of time to process randomized latency. In the case of receiving MH-AIT after this time, the control code is applied at once. Date is coded by the lower 16 bits of MJD, and as for time, hour, minute and second of the Japan Standard Time (JST) are coded by BCD 24 bits.

10.3.3.8 MH-External Application Control Descriptor

The MH-external application control descriptor specifies access authority to broadcasting resource which is given to an external application. This descriptor can be arranged at most one in the common descriptor loop of MH-AIT. Working in the case where it is not arranged is provided by operation. Also, the control information for the AIT-controlled application may be arranged in the common descriptor loop of MH-AIT in which this descriptor is arranged and in the application information descriptor loop. The configuration of the MH-external application control descriptor is shown in Table 10-22.

Table 10-22 Configuration of MH-external application control descriptor

Data structure	Number of bit	Data notation
MH-External_Application_Control_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
specific_scope_flag	1	bslbf
reserved_for_future_use	7	bslbf
if(specific_scope_flag == 1){		
target_application_class	16	bslbf
target_application_count	8	uimsbf
for(i=0; i<target_application_count; i++){		
target_application(){		
application_identifier()		
}		
}		
}		
permission_bitmap_count	8	uimsbf
for(i=0; i<permission_bitmap_count; i++){		
permission_bitmap	16	bslbf
}		
overlay_admission_polarity	1	bslbf
reserved_for_future_use	3	bslbf
overlay_controlled_area_count	4	uimsbf
for(i=0; i<overlay_controlled_area_count; i++){		
overlay_controlled_area_tag	8	uimsbf
horizontal_pos	16	uimsbf
vertical_pos	16	uimsbf
horizontal_size	16	uimsbf
vertical_size	16	uimsbf
}		
blocked_application_count	8	uimsbf
for(i=0; i< blocked_application _count; i++){		
blocked_application(){		
application_identifier()		
}		
}		
}		

The meaning of MH-external application control descriptor:

specific_scope_flag (specific scope designation flag): When this field is '0', this represents all applications shall be the target for setting the access authority, and when it is '1', it represents that the setting of the target application class and the application specified by setting the target application shall be the target for the setting of the access authority. Plural descriptors of this can be arranged, and among these descriptors which the concerned application is the target for setting, the direction of those arranged in the forefront is valid. So, when plural descriptors of this are arranged, it is necessary to consider so that those which this field is '0' are arranged in

the end of the descriptors, etc.

target_application_class (target application class): This represents the class of the application which is a target for setting the access authority. It represents when '1' is set to each bit, the application of the concerned class becomes a target for setting the access authority. When '1' is set to plural bits, the application which belongs to any of those classes becomes a target for setting the access authority.

target_application_count (target application count): The number of applications. The number of applications for which the access authority is set.

target_application() (target application): The identification information of the application which is the target for setting the access authority. This is described in the application identification provided in the ARIB STD-B62. Also, in the case that the application which is represented by this application identification matches any of the application class ranges, the target for setting the access authority shall be designated.

permission_bitmap_count (permission bitmap count): The number of the access authority bitmaps.

permission_bitmap (permission bitmap): It is configured by a bitmap of 16 bits for every functions whether access to each broadcasting resource is permitted or not. The upper three bits represents the change of the bitmap. The assignment of the functional bitmap is provided in operation. When it is judged that it does not have the access authority, the receiver must disable the access.

overlay_admission_polarity (video overlay admission polarity): This represents whether overlay is permitted or prohibited in the overlay control area which is designated by the following field. If it is permitted, it shall be '1', and if it is prohibited, it shall be '0'.

overlay_controlled_area_count (video overlay controlled area count): The number of video overlay controlled area.

overlay_controlled_area_tag (video overlay controlled area tag): The identification number of rectangle video overlay area which is specified in the following.

horizontal_pos (video overlay controlled area horizontal position): The horizontal coordinate in the upper and left corners of the video overlay area. This is represented by the number of pixels.

vertical_pos (video overlay controlled area vertical position): Vertical coordinate in the upper and left corners of the video overlay area. This is represented by the number of pixels.

horizontal_size (video overlay controlled area horizontal size): Width of the video overlay area. This is represented by the number of pixels.

vertical_size (video overlay controlled area vertical size): Height of the video overlay area. This is represented by the number of pixels.

blocked_application_count (blocked application count): This represents the number of the application in which broadcasting resource is absolutely prohibited to access as a black list which is specified in the following.

blocked_application() (blocked application): This represents the application which is a target to be absolutely prohibited to access broadcasting resources. This is described in the application identification provided in the ARIB STD-B62.

10.3.3.9 MH-Playback Application Descriptor

The MH-playback application descriptor arranges the application which starts up with

playback of recorded content in the application information descriptor loop of MH-AIT, in the case they want to make the application different from the application which starts up with live viewing of the concerned content. When this descriptor is not arranged in MH-AIT, the application with playback of recorded content also starts up according to the designation of the MH-application descriptor and the MH-simple application location descriptor. In the case of designating applications by using this descriptor, recorded content to be played back is treated as those are equivalent to one service. Therefore, "present service" means the content being played back (involving data which is multiplexed in the program service). The configuration of the MH-playback application descriptor is shown in Table 10-23.

Table 10-23 Configuration of MH-playback application descriptor

Data structure	Number of bit	Data notation
MH-Playback_Application_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
application_profiles_length	8	uimsbf
for(i=0; i<N; i++){		
application_profile	16	uimsbf
version.major	8	uimsbf
version.minor	8	uimsbf
version.micro	8	uimsbf
}		
service_bound_flag	1	bslbf
visibility	2	bslbf
reserved_for_future_use	5	bslbf
application_priority	8	uimsbf
for(i=0; i<N; i++){		
transport_protocol_label	8	uimsbf
}		
}		

The meaning of MH-playback application descriptor:

application_profiles_length (application profile information length): This represents the byte length of the whole application profile information involved in the following loop.

application_profile (application profile): This represents application profile of the receiver which this application can be executed. If the receiver implements this profile, it shows that it has a faculty to execute this application. The content of profile is defined by each application format, respectively.

version.major (major version): This represents major version of the above-mentioned profile.

version.minor (minor version): This represents minor version of the above-mentioned profile.

version.micro (micro version): This represents micro version of the above-mentioned profile.

Minimum profile to execute this application is represented by the above-mentioned four fields. In the case that a profile fits at least one logical operation shown in the following and there is a profile which results in true in this application profile information, the receiver shall startup this application.

(profile of application ∈ a set of profile which is implemented in the terminal)
AND { (major version of application < major version of the terminal for this profile)
OR [(major version of application = major version of the terminal for this profile))
AND ({ minor version of application < minor version of the terminal for this profile)}
OR { [minor version of application = minor version of the terminal for this profile)]
AND [micro version of application ≦ micro version of the terminal for this profile)]})] }

service_bound_flag (service bound flag): This represents whether this application is valid only for present service or not. In the case that this field is '1', the application is connected to only present service, and in the case of changing to the other service, the application will finish.

visibility (visibility): This represents whether visible or not for users and the other application in executing this application. Refer to the Table 10-24.

Table 10-24 Visibility

Value of visibility	meaning
00	This application shall be invisible except reporting error such as log output.
01	This application shall be invisible to users, but visible from the other application through API, etc.
10	reserved_future_use
11	This application shall be visible to either user or the other application.

application_priority (application priority): This represents the relative priority among applications in the case that plural applications work.

transport_protocol_label (transport protocol label): This represents the value whch uniquely identifies the transport protocol which is transmitting the application. It corresponds to the field with the same name as the MH-Transport protocol descriptor which is recorded in video recording and is acquired at playback.

10.3.3.10 MH-Simple Playback Application Location Descriptor

The MH-simple playback application location descriptor designates details of the acquisition destination which starts up at playback of recorded content. When the MH-playback application descriptor is arranged in MH-AIT, one MH-simple playback application location descriptor per one MH-playback application descriptor shall be always arranged in the same loop. The configuration of the MH-simple playback application location descriptor is shown in Table 10-25.

Table 10-25 Configuration of MH-simple playback application location descriptor

Data structure	Number of bit	Data notation
MH-Simple_Playback_Application_Location_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for(i=0; i<N; i++){		
initial_path_bytes	8	uimsbf
}		
}		

The meaning of MH-simple playback application location descriptor:

initial_path_bytes (application URL): Character string which represents the URL of the entry point of corresponding application. It is represented by the relative path where acquirable location of the application represented by the MH-tranport protocol descriptor is a root.

10.3.3.11 MH-Application Expiration Descriptor

The MH-application expiration descriptor designates the term of validity that the application which is directed by the MH-payback application descriptor and the MH-simple playback application location descriptor can be started up. When the MH-playback application descriptor is arranged, at most one MH-application expiration descriptor per one MH-playback application descriptor can be arranged in the same loop. When this descriptor is arranged and the time of playback passes expiration which this descriptor designates, the receiver must not startup the concerned application. In the case that this descriptor is not arranged, the concerned application may be started up in spite of the time of playback. The configuration of the MH-application expiration descriptor is shown in Table 10-26.

Table 10-26 Configuration of MH-application expiration descriptor

Data structure	Number of bit	Data notation
MH-Application_Expiration_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
expiration_date_and_time	40	bslbf
}		

The meaning of MH-application expiration descriptor:

expiration_date_and_time (expiration date and time): This is the value representig date and time of expiration date for the application. It is denoted by the Modified Julian Day (MJD) and the Japan Standard Time (JST), and the lower 16 bits of MJD is coded by 16 bits, and the subsequent 24 bits are coded by six binary coded decimals (BCD) of 4 bits.

10.3.4 Descriptor Used in Data Asset Management Table

10.3.4.1 MH-Type Descriptor

The MH-Type descriptor represents the type of the file which is transmitted in the application transmission system. The configuration of the MH-Type descriptor is shown in Table 10-27.

Table 10-27 Configuration of MH-type descriptor

Data structure	Number of bit	Data notation
MH-Type_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
for (i=0; i<N; i++) {		
text_char	8	uimsbf
}		
}		

The meaning of MH-Type descriptor:

text_char (media description): This is a field of 8 bits, and a series of regions represents media types based on RFC1521 and RFC1590. The designation method of the media type is specified for each application range, as the following.

- Multimedia coding based on XML: ARIB STD-B24 Fascicle 2, Annex, Provision C
- Conditional access system: ARIB STD-B25 Part 2
- Server type broadcasting system: ARIB STD-B38
- Media Transport System by MMT: ARIB STD-B60

10.3.4.2 MH-Info Descriptor

The MH-Info descriptor describes the information about MPU or item. The configuration of the MH-Info descriptor is shown in Table 10-28.

Table 10-28 Configuration of MH-info descriptor

Data structure	Number of bit	Data notation
MH-Info_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
ISO_639_language_code	24	bslbf
for (i=0; i<N; i++) {		
text_char	8	uimsbf
}		
}		

The meaning of MH-Info descriptor:

ISO_639_language_code (language code): This field of 24 bits identifies language in the following text_char region. The language code is represented by alphabetical three letters code provided in the ISO639-2. Each letter is coded according to the ISO8859-1 by 8 bits, and is

inserted in 24 bits field by the order.

text_char (information description): This is a field of 8 bits. A series of region represents a character string about the file which is transmitted as item by using the character code provided in the data coding system or in operation.

10.3.4.3 MH-Expire Descriptor

The MH-Expire descriptor describes the period of validity for item. For the example, in the case that the item is stored in the receiver which has a storage device, the stored data will be deleted at the expiration. The item in which this descriptor is not described means that expiration is not set. The configuration of the MH-Expire descriptor is shown in Table 10-29.

Table 10-29 Configuration of MH-expire descriptor

Data structure	Number of bit	Data notation
MH-Expire_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
time_mode	8	uimsbf
if (time_mode == 0x01) {		
UTC_time	64	uimsbf
}		
else if (time_mode == 0x04) {		
reserved_future_use	8	bslbf
passed_seconds	32	uimsbf
}		
}		

The meaning of MH-expire descriptor:

time_mode (time mode): The way to specify the expiration time is shown in Table 10-30.

Table 10-30 Time mode

time_mode	Method to designate time	meaning
0x00	-	Reserved for future use
0x01	UTC_time	Absolute time represented by UTC
0x02	-	Reserved for future use
0x03	-	Reserved for future use
0x04	passed_seconds	Lapsed time after download (second)
0x05-0xFF	-	Reserved for future use

UTC_time (UTC time): This field of 64 bits is coded in the case that time_mode=0x01, and the expiration is denoted as the absolute time of Coordinated Universal Time (UTC) by the NTP time stamp format. When the MSB of 32 bits which represents the unit of second is 0, the year 2036 shall be the reference.

passed_seconds (lapsed seconds): This field of 32 bits is coded in the case that time_mode=0x04, and the expiration is denoted as the passed time (unit: second) after download.

10.3.4.4 MH-Compression Type Descriptor

The MH-compression type descriptor means the item to be transmitted is compressed, and represents the compression algorithm and the number of byte of the item before compression. This descriptor is not given to the item which is not compressed. The configuration of the MH-Compression type descriptor is shown in Table 10-31.

Table 10-31 Configuration of MH-compression type descriptor

Data structure	Number of bit	Data notation
MH-Compression_Type_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
compression_type	8	uimsbf
original_size	32	uimsbf
}		

The meaning of MH-Compression type descriptor:

compression_type (compression identification): This field of 8 bits specifies the compression format which is used for compression of the item. The value to identify the compression format is provided in opration.

original_size (original size): This field of 32 bits represents the size of the item before compression by the number of bytes.

10.3.4.4 MPU Node Descriptor

The MPU node descriptor is arranged in mpu_info_byte, and represents the concerned MPU corresponds to the directory node which is provided in the data directory management table. The configuration of the MPU node descriptor is shown in Table 10-32.

Table 10-32 Configuration of MPU node descriptor

Data structure	Number of bit	Data notation
MPU_Node_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
node_tag	16	uimsbf
}		

The meaning of MPU node descriptor:

node_tag (node tag): This represents the label which identifies the directory, as a node tag of

the directory node which corresponds to the concerned MPU.

10.3.5 Descriptor Used in Data Content Management Table

10.3.5.1 Linked PU Descriptor

The linked PU descriptor represents the other presentation unit to which has a possibility to transit from the concerned presentation unit. This functions as a hint information in order that the receiver previously pre-caches the file of linked presentation unit. The configuration of the linked PU descriptor is shown in Table 10-33.

Table 10-33 Configuration of linked PU descriptor

Data structure	Number of bit	Data notation
Linked_PU_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
num_of_linked_PU	8	uimsbf
for(i=0; i<num_of_linked_PU; i++){		
linked_PU_tag	8	uimsbf
}		
}		

The meaning of linked PU descriptor:

num_of_linked_PU (number of linked presentation units): This represents the number of linked presentation unit designation.

linked_PU_tag (linked presentation units tag): This represents the presentation unit tag of the linked presentation unit.

10.3.5.2 Locked Cache Descriptor

The locked cache descriptor represents the file of target to cache in the concerned presentation unit and to fix (lock) it. In the case that this descriptor exists, after acquiring files which were not acquired, the receiver needs to fix the file of target in cache memory. The file of target for locking must be unlocked or held on cache memory until the end of application. The configuration of the locked cache descriptor is shown in Table 10-34.

Table 10-34 Configuration of locked cache descriptor

Data structure	Number of bit	Data notation
Locked_Cache_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
num_of_locked_cache_node	8	uimsbf
for(i=0; i<num_of_locked_cache_node; i++){		
node_tag	16	uimsbf
}		
}		

The meaning of locked cache descriptor:

num_of_locked_cache_node (number of locked cache node): This represents the number of nodes which are targets for locking.

node_tag (node tag): This represents the file of target for locking or the node tag of directory.

10.3.5.3 Unlocked Cache Descriptor

The unlocked cache descriptor represents the target to dissolve (unlock) fixation among files which are fixed (locked) in the concerned presentation unit. In the case that this descriptor exists, the receiver may drop designated files from the target for locking, and may delete them when caching the other file. The configuration of the unlocked cache descriptor is shown in Table 10-35.

Table 10-35 Configuration of unlocked cache descriptor

Data structure	Number of bit	Data notation
Unlocked_Cache_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
num_of_unlocked_cache_node	8	uimsbf
for(i=0; i<num_of_unlocked_cache_node; i++){		
node_tag	16	uimsbf
}		
}		

The meaning of unlocked cache descriptor:

num_of_unlocked_cache_node (number of unlocked cache node): This represents the number of nodes which are targets for unlocking.

node_tag (node tag): This represents the file of target for unlocking or the node tag of directory.

10.3.5.4 PU Structure Descriptor

The PU structure descriptor represents the list of the MPU configuring presentation unit, as mapping the information of the concerned presentation unit and the unit of transmission. The configuration of the PU structure descriptor is shown in Table 10-36.

Table 10-36 Configuration of PU structure descriptor

Data structure	Number of bit	Data notation
PU_Structure_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
num_of_MPU	8	uimsbf
for(i=0; i<num_of_MPU; i++) {		
MPU_sequence_number	32	uimsbf
}		
}		

The meaning of PU structure descriptor:

num_of_MPU (number of MPU): This represents the number of MPU configuring the concerned presentation unit.

MPU_sequence_number (MPU sequence number): This represents the sequence number of MPU configuring the concerned presentation unit.

Chapter 11: Transmission of Event Message

11.1 Summary of Event Message Transmission System

The event message transmission system supplies a means to send message informations from a broadcasting station to the application which is working in the receiver, immediately or at the designated time. The event message transmission system that is provided in the ARIB STD-B24 is an extention of the stream descriptor defined in the ISO/IEC 13818-6 and the transmission specification by DSM-CC, to use various time designation systems which are required by application. The same specification as this is transmitted by the M2 section message in the event message system.

11.2 Control Information in Event Message Transmission System

11.2.1 Table Used in Event Message Transmission System

11.2.1.1 Event Message Table (EMT)

The event message table is used to store the UTC-NPT reference descriptor or the event message descriptor and to transmit the information about the event message. The event message table is transmitted by storing it in the M2 section message. The configuration of the event message table is shown in Table 11-1.

Table 11-1 Configuration of event message table

Data structure	Number of bit	Data notation
Event_Message_Table (){		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
reserved_future_use	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
data_event_id	4	uimsbf
event_msg_group_id	12	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for(i=0; i<N; i++) {		
descriptor ()		
}		
CRC_32	32	rpchof
}		

The meaning of event message table:

section_syntax_indicator (section syntax indicator): This field of one bit represents that CRC32 exists at the end of section in the case of '1', and check-sum exists in the case of '0'. It shall be always '1' in the transmission of the event message.

section_length (section length): This field of 12 bits represents the byte length from immediately after this field to the end of section. The value of this field does not exceed 4093.

data_event_id (data event identification): This field of 4 bits is an identifier aimed to discriminate data events which are next to each other in the time to use the event message, and to avoid misdistributing the event message between local contents which are distributed in these data events. The different identifiers are assigned between adjacent local contents when transmitting.

event_msg_group_id (message group identification): This field of 12 bits represents an identifier which identifies a message group to be received by the application. Detail operation is specified for each data coding system identification.

version_number (version number): This field of 5 bits is the version number of the sub-table. One is added to the version number when there is a change in the information in the sub-table. But when the present number is 31, next version number will return to 0.

current_next_indicator (current next indicator): This indication of one bit represents that the sub-table is the present sub-table in the case that it is '1'. In the case that it is '0', it represents that the sub-table to be transmitted is not applied yet, and is used as the next sub-table.

section_number (section number): This field of 8 bits represents the section number.

last_section_number (last section number): This field of 8 bits represents the last section (that is, the section with maximum number) number of the sub-table which the section belongs to.

11.2.2 Descriptor Used in Event Message Transmission System

11.2.2.1 UTC-NPT Reference Descriptor

The UTC-NPT reference descriptor is a descriptor for transmitting relationship between the NPT (Normal Play Time) and the UTC (Universal Time Coordinated). The configuration of the UTC-NPT reference descriptor is shown in Table 11-2.

Table 11-2 Configuration of UTC-NPT reference descriptor

Data structure	Number of bit	Data notation
UTC-NPT_Reference_Descriptor (){		
descriptor_tag	16	uimsbf
descriptor_length	8	uimsbf
UTC_Reference	64	uimsbf
NPT_Reference	64	uimsbf
UTC_NPT_ leap_indicator	2	uimsbf
scale	2	bslbf
reserved	4	bslbf
}		

The meaning of UTC-NPT reference descriptor:

UTC_Reference (UTC reference time): In this field of 64 bits, the value of the universal time (UTC) corresponding to the value of the NPT_Reference is stored.

NPT_Reference (NPT reference time): This field of 64 bits represents the value of NPT when the universal time (UTC) becomes the value of the UTC_Reference.

UTC_NPT_leap_indicator (UTC_NPT leap second indicator): When leap second is adjusted to the system clock of the transmission system, in the case that the UTC reference time is from 9:00:00 am on the day before the leap second insertion day in the Japan time to first 8:59:59 am on the leap second insertion day, this represents 1. In the case that the UTC reference time is from 9:00:00 am on the day before the leap second deletion day to 8:59:58 am on the leap second deletion day, this represents 2, and in other case, this represents 0. 3 shall be reserved.

scale (scale): This field of two bits represents the rate which the NPT advances. In the case that it is 01, The NPT keeps a fixed value regardless of the UTC. In the case that it is 11, the NPT advances in the same rate as the UTC.

11.2.2.2 Event Message Descriptor

The event message descriptor is used for transmitting the information about event messages in general. The data structure of the event message descriptor is shown in Table 11-3.

Table 11-3 Configuration of event message descriptor

Data structure	Number of bit	Data notation
Event_Message_Descriptor () {		
descriptor_tag	16	uimsbf
descriptor_length	16	uimsbf
event_msg_group_id	12	uimsbf
reserved_future_use	4	bslbf
time_mode	8	uimsbf
if (time_mode == 0) {		
reserved_future_use	64	bslbf
} else if (time_mode == 0x01 || time_mode==0x05) {		
event_msg_UTC	64	uimsbf
} else if (time_mode == 0x02) {		
event_msg_NPT	64	uimsbf
} else if (time_mode == 0x03) {		
event_msg_relativeTime	64	uimsbf
}		
event_msg_type	8	uimsbf
event_msg_id	16	uimsbf
for (i=0; i<N; i++) {		
private_data_byte	8	uimsbf
}		
}		

The meaning of event message:

event_msg_group_id (message group identification): This field of 12 bits represents the identifier that identifies message groups to be received by the application. Detail operation is specified for each data coding identification. Also, in the case of operating an event message which has the plural message group identifications at the same time, it shall involve only general-purpose event message that has the same message group identification in one M2 section.

time_mode (time mode): This field of 8 bits represents how to specify the time when event message occurs. Refer to Table 11-4.

Table 11-4 Configuration of time mode

time_mode	Method of time designation	meaning
0x00	none	Event message is generated at once after receiving.
0x01	UTC	Event message is generated according to absolute time which is represented by UTC. But when content is reproduced which is recorded by stream, event message is also generated by referring to the time of reproduction.
0x02	NPT	Event message is generated according to NPT time data.
0x03	relativeTime	Event message is generated according to relative time which is represented by UTC.
0x04	-	Reserved for use in the future.
0x05	UTC	Event message is generated according to absolute time which is represented by UTC. But when content is reproduced which is recorded by stream, event message is generated by referring to the time of broadcasting.
0x06-0xFF	-	Reserved for use in the future

event_msg_UTC (message UTC time): This field of 64 bits is coded in the case that the time mode is 0x01 or 0x05, and represents the time when the event message occurs as the absolute time of the Coordinated Universal Time (UTC) by the NTP time stamp format. When the MSB of 32 bits which represents the unit of second is 0, the year 2036 shall be the standard.

event_msg_NPT (message NPT time): This field of 64 bits is coded in the case that the time mode is 0x02, and represents the time when the event message occurs by the Normal Play Time which is represented by the UTC-NPT reference descriptor.

event_msg_relativeTime (message relative time): This field of 64 bits is coded in the case that the time mode is 0x03, and represents the time when the event message occurs by relative time from the start time of a program. The relative time is represented by the difference of the NTP time stamp format of the Coordinated Universal Time (UTC).

event_msg_type (message type): This is an identifier which represents the kind of the event message. The operation and the meaning are specified for each data coding system.

event_msg_id (message identification): This field of 16 bits represents an identifier to identify each event message. The operation and the meaning are specified for each data coding system.

private_data_byte (private data): This is a field of 8 bits. In a series of regions, related information which is defined according to the value of event_msg_type which are specified for each data coding system is stored.

Chapter 12: Transmission of General-purpose Data

12.1 Summary of General-purpose Data Transmission System

The general-purpose data transmission system is a system for transmitting various kinds of data in a broadcasting system which uses MMT. There are two systems in general-purpose data transmission system defined in this chapter; one is the system that transmits by capsulizing in synchronous type MPU, and the other is the system that transmits by capsulizing in asynchronous type MPU. They are called as the synchronous type general-purpose data transmission system and the asynchronous type general-purpose data transmission system, respectively. The synchronous type general-purpose data transmission system is the system which is used when it is necessary to synchronize timing between transmitted data and the other stream such as video, audio and so on. The asynchronous type general-purpose data transmission system is the system which is used when it is not necessary to synchronize timing between transmitted data and the other stream such as video, audio and so on.

The general-purpose data transmission can be used for transmission of the data which is used by player that presents the data other than video, audio and closed-caption, or used for the data streaming for the HTML5 application. For the example, it is assumed that transmitting picture data by the synchronous type general-purpose data transmission system, the service in which video is presented by precise synchronization is offered by the picture presentation player which is implemented in the receiver, and push-type data is applied to streaming for the HTML5 application by the asynchronous type general-purpose data transmission system.

The general-purpose data is transmitted by capsulizing in the synchronous type MPU or the asynchronous type MPU, and the control information about transmission is transmitted by the MP table. Overview of acquisition of the general-purpose data is shown in Fig. 12-1.

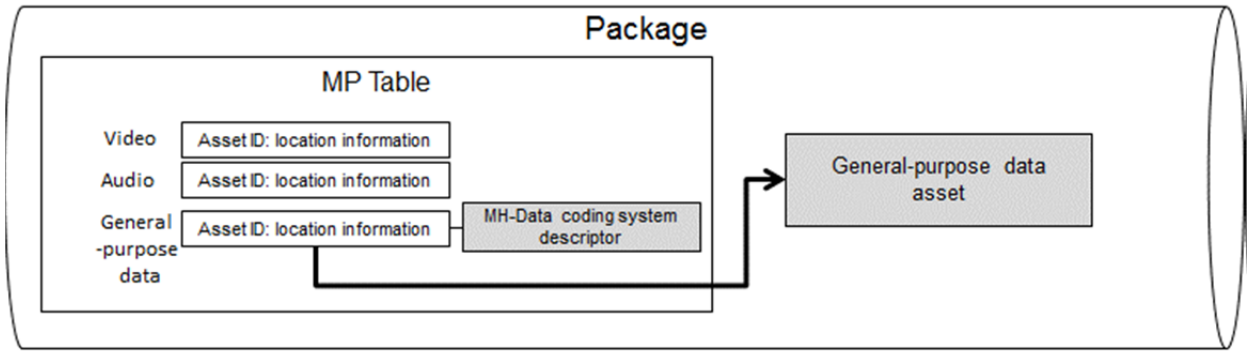


Fig. 12-1 Overview of general-purpose data acquisition

The receiver refers to the MP table, and detects whether there are assets which transmits the general-purpose data or not. In the case that the general-purpose data is transmitted, there is the asset that the value which represents the synchronous type general-purpose data transmission system or the asynchronous general-purpose data transmission system is denoted in the asset type, and the MH-data coding system descriptor is arranged in the asset descriptor region. By referring to the MP table, the location information of asset is identified, and the packet ID which transmits the general-purpose data is specified. By expanding the data in MPU which is transmitted in the asset of the specified packet ID, the file that is transmitted as the general-purpose data can be acquired.

12.2 General-purpose Data Transmission System

In the general-purpose data transmission system, the data is transmitted by using the synchronous type MPU (MPU with Timed Media Data) and the asynchronous type MPU (MPU with non-Timed Media Data) which are defined in the MMT standard. The general-purpose data which is transmitted by the synchronous type MPU is called as the synchronous type general-purpose data, and the general-purpose data which is transmitted by the asynchronous type MPU is called as the asynchronous type general-purpose data. The presentation time is shown to the MPU for the synchronous type general-purpose data by the MPU timestamp descriptor.

12.2.1 Configuration of MFU/MPU for General-purpose data

The MPU is the unit of processing in the processing of the general-purpose data transmission. The MPU involves data which is used at a certain timing, and is the unit that the presentation and so on can be processed by the unit of the MPU. The configuration of the MPU/MFU for the general-purpose data is shown in Fig. 12-2.

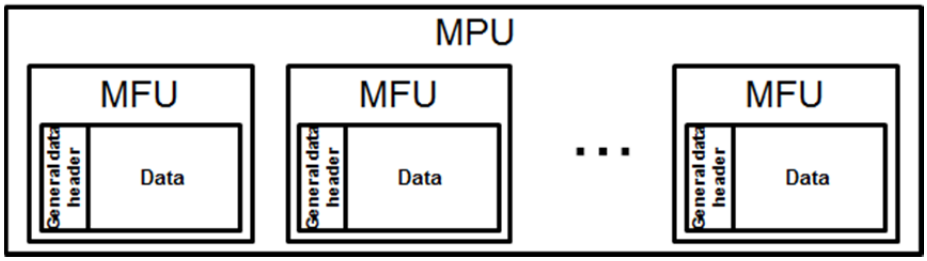


Fig. 12-2 Configuration of MPU/MFU for general-purpose data

In the MPU for the general-purpose data, the MPU metadata and the movie fragment metadata are not arranged, but it is configured by more than or equal to one MFU. Each MFU is composed of the general-purpose data header part and the data part. In the data part of each MFU, the data with an arbitrary size can be stored, for the example, one file can be transmitted by one MFU.

12.3 Configuration of MFU for General-purpose Data

The configuration of the MFU for the general-purpose data is shown in Table 12-1.

Table 12-1 Configuration of MFU for general-purpose data

Data structure	Number of bit	Data notation
MFU_data_byte() {		
component_tag	16	uimsbf
data_type_length	8	uimsbf
for (i=0; i<N; i++) {		
data_type_char	8	char
}		
additional_info_length	16	uimsbf
for (j=0; j<N; j++) {		
additional_info_byte	8	bslbf
}		
data_length	32	uimsbf
for (k=0; k<N; k++) {		
data_byte	8	bslbf
}		
}		

The meaning of MFU for general-purpose data:

component_tag (component tag): The value of component tag for the general-purpose data which is given by the MH-stream identification descriptor of the MPT is put in this field of 16 bits.

data_type_length (data type length): This represents the length of the following data type description by the unit of byte.

data_type_byte (data type byte): This shall be the region where the MIME type of data which is transmitted by the concerned MFU is stored.

additional_info_length (additional information byte): This represents the length of the following additional information description by the unit of byte.

additional_info_byte (additional information byte): This shall be the region where additional information about data which is transmitted by the concerned MFU is stored.

data_length (data length): This represents the length of data byte of the following general-purpose data by the unit of byte.

data_byte (data byte): This is the region where the general-purpose data is stored.

12.4 Descriptor in General-purpose Data

As for the data which is transmitted by using the general-purpose data transmission system, the MH-data coding system descriptor is arranged in the MP table and the data_component_id is assigned, and also the MH-stream descriptor is arranged and the component tag value is assigned to each asset.

Annex

Annex 1 Assignment Method of Identifier

1 Assignment Methods of Identifier

The assignment methods of identifier which are used for TLV multiplex system and MMT are based on Table A1-1 and Table A1-2, respectively.

The range of values in the tables includes undefined values which are possible to be allocated in the future. Therefore, the provision of broadcasters may leave undefined, but the signals of broadcasters including undefined are registered and opened. The signals of broadcasters shall be valid only in the network identification of the concerned broadcasters. But plural broadcasters can also use the signals of broadcasters by the same format.

Table A1-1 Assignment method of identifiers used in TLV multiplex system

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Table identification*1 (table_id)	8	4.1	0x00 – 0x41, 0x82 – 0x86, 0xFF	Provided by MIC	Provided in Notification
			0x42 – 0x81, 0x87 – 0x8F, 0xC0 – 0xFE	Provided by standardization organization	Registered and opened after discussion
			0x90 – 0xBF	Provided and operated by broadcaster	
Network identification*1 (network_id)	16	5.2.1.1 B.1	all	Provided by standardization organization	Registered by application and opened
Original network identification*1 (original_network_id)					
TLV stream identification (tlv_stream_id)	16	5.2.1.1	all	Provided and operated by broadcaster	Unique within original network identification
Descriptor tag*1 (descriptor_tag)	8	4.1	0x00 – 0x3F, 0x41, 0x43, 0x44, 0xF5 – 0xFE	Provided by MIC	Provided in Notification
			0x40, 0x42, 0x45 – 0x7F, 0xC0 – 0xF4, 0xFF	Provided by standardization organization	Registered and opened after discussion
			0x80 – 0xBF	Provided and operated by broadcaster	

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Service identification (service_id)	16	5.3.1.1	all	Provided and operated by broadcaster	Unique in Japan in the same broadcasting media
System management identification*1 (system_management_id)	2+6+8	5.3.1.3	8 bits in higher rank	Provided by MIC	Provided in Notification
			8 bits in lower rank	Provided and operated by broadcaster	
Remote controller key identification (remote_control_key_id)	8	5.3.1.5	all	Provided and operated by broadcaster	

*1 : Identifier whose value is assigned in the same range as the provision in the ARIB STD-B10

Table A1-2 Assignment method for identifier used in MMT

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Table identification (table_id)	8	4.2	0x00 – 0x10, 0x21 – 0x81, 0x87 – 0xDF	Provided by standardization organization	Registered and opened after discussion
			0x11 – 0x20, 0x82 – 0x86	Provided by MIC	Provided in Notification
			0xE0 – 0xFF	Provided and operated by broadcaster	
Descriptor tag (descriptor_tag)	16	4.2	0x0001, 0x0002, 0x8004, 0x8005, 0x8007	Provided by MIC	Provided in Notification
			0x0000, 0x0003 – 0x8003, 0x8006, 0x8008 – 0xEBFF 0xF000 – 0xFBFF	Provided by standardization organization	Registered and opened after discussion
			0xEC00 – 0xEFFF, 0xFC00 – 0xFFFF	Provided and operated by broadcaster	
Item identification (item_id)	32	6.3.2 7.4.3.36	all	Provided and operated by broadcaster	
FEC type (FEC_type)	2	6.4.1.1	all	Provided by MIC	Provided in Notification
Payload type (payload_type)	6	6.4.1.1	0x00 – 0x02	Provided by MIC	Provided in Notification
			0x03 – 0x3F	Provided by standardization organization	Registered and opened after discussion
Packet identification (packet_id)	16	4.3 6.4.1.1	0x0000, 0x0001	Provided by MIC	Provided in Notification
			0x0002 – 0x8FFF	Provided by standardization organization	Registered and opened after discussion
			0x9000 – 0xFFFF	Provided and operated by broadcaster	
Extension header type (extension_type)	16	6.4.1.1	all	Provided by standardization organization	Registered and opened after discussion
Multi-extension header type (hdr_ext_type)	15	6.4.1.1	all	Provided by standardization organization	Registered and opened after discussion

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Message identification (message_id)	16	4.2 7.2.3	0x0000, 0x8000, 0x8001	Provided by MIC	Provided in Notification
			0x0001 – 0x7FFF, 0x8002 – 0xDFFF, 0xF000 – 0xF7FF	Provided by standardization organization	Registered and opened after discussion
			0xE000 – 0xEFFF, 0xF800 – 0xFFFF	Provided and operated by broadcaster	
Package ID byte (MMT_package_id _byte)	variable length	7.3.3.1	all	Provided and operated by broadcaster	Lower 16 bits are the same value as service identification
Asset ID byte (asset_id_byte)	variable length	7.3.3.1	all	Provided and operated by broadcaster	
Transport stream identifier (MPEG_2_transport _stream_id)	16	7.3.3.1	all	Provided and operated by broadcaster	Unique within original network identification
MPEG-2 packet identifier*1 (MPEG_2_PID)	13	7.3.3.1	0x0000 – 0x0010, 0x1FFF	Provided by MIC	Provided in Notification
			0x0011 – 0x002F	Provided by standardization organization	Registered and opened after discussion
			The value which does not interfere with the abobe-mentioned	Provided and operated by broadcaster	
Transport file identification (transport_file_id)	32	7.3.3.2	all	Provided and operated by broadcaster	Unique within original network identification
Layout number (layout_number)	8	7.3.3.3 7.4.3.4	0	Provided by standardization organization	Registered and opened after discussion
			1 – 255	Provided and operated by broadcaster	
Device ID (device_id)	8	7.3.3.3	0	Provided by standardization organization	Registered and opened after discussion

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
			1 – 255	Provided and operated by broadcaster	
Region number (region_number)	8	7.3.3.3 7.4.3.4	0	Provided by standardization organization	Registered and opened after discussion
			1 – 255	Provided and operated by broadcaster	
Event identification (event_id)	16	7.3.3.9	all	Provided and operated by broadcaster	
Download data identification (download_data_id)	16	7.3.3.10 7.4.3.34	all	Provided and operated by broadcaster	
Data type (data_type)	8	7.3.3.10	all	Provided by standardization organization	Registered and opened after discussion
Broadcaster identification (broadcaster_id)	8	7.3.3.11 他	all	Provided and operated by broadcaster	
Group ID (group_identification)	8	7.4.3.1	all	Provided and operated by broadcaster	
Selection level (selection_level)	8	7.4.3.1	0	Provided by standardization organization	Registered and opened after discussion
			1 – 255	Provided and operated by broadcaster	
Conditional access system identifier*1 (CA_system_ID)	16	7.4.3.7 B.2	all	Provided by standardization organization	Registered and opened after discussion
Scramble system identifier (scramble_system_id)	8	7.4.3.8	all	Provided by standardization organization	Registered and opened after discussion
Message authentication system identifier (message_authentication_id)	8	7.4.3.9	all	Provided by standardization organization	Registered and opened after discussion
Area code*1 (area_code)	12	7.4.3.10	all	Provided by MIC	Provided in Notification

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Linkage type*1 (linkage_type)	8	7.4.3.14	0x00 – 0x7F, 0xC0 – 0xFF	Provided by standardization organization	Registered and opened after discussion
			0x80 – 0xBF	Provided and operated by broadcaster	
Group type*1 (group_type)	4	7.4.3.15	all	Provided by standardization organization	Registered and opened after discussion
Service type*1 (service_type)	8	7.4.3.16	0x00 – 0x7F, 0xC0 – 0xFF	Provided by MIC	Provided in Notification
			0xA1 – 0xBF	Provided by standardization organization	Registered and opened after discussion
			0x80 – 0xA0	Provided and operated by broadcaster	
Component tag (component_tag)	16	7.4.3.20	all	Provided and operated by broadcaster	
Genre 1*1 (content_nibble_level_1)	4	7.4.3.21	all	Provided by standardization organization	Registered and opened after discussion
Genre 2*1 (content_nibble_level_2)	4	7.4.3.21	all	Provided by standardization organization	Registered and opened after discussion
User genre*1 (user_nibble)	4+4	7.4.3.21	all	Provided and operated by broadcaster	
Age limitation rating*1 (rating)	8	7.4.3.22	0x00 – 0x0F	Provided by standardization organization	Registered and opened after discussion
			0x10 – 0xFF	Provided and operated by broadcaster	
Component content*1 (stream_content)	4	7.4.3.23	0x0 – 0xB	Provided by standardization organization	Registered and opened after discussion
			0xC – 0xF	Provided and operated by broadcaster	
Component type (component_type)	1+2+5	7.4.3.23	all	Provided by standardization organization	Registered and opened after discussion

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Stream type*1 (stream_type)	8	7.4.3.23	0x00 – 0x7F	Provided by MIC	Provided in Notification
			0x80 – 0xC3	Provided by standardization organization	Registered and opened after discussion
			0xC4 – 0xFF	Provided and operated by broadcaster	
Simulcast group identification*1 (simulcast_group_tag)	8	7.4.3.23	0x00 – 0xFE	Provided and operated by broadcaster	
			0xFF	Provided by standardization organization	Registered and opened after discussion
Region description system designation*1 (region_spec_type)	8	7.4.3.24	all	Provided by standardization organization	Registered and opened after discussion
Series identification (series_id)	16	7.4.3.25	all	Provided and operated by broadcaster	Unique within broadcaster identification
Repeat label*1 (repeat_label)	4	7.4.3.25	0x0	Provided by standardization organization	Registered and opened after discussion
			0x1 – 0xF	Provided and operated by broadcaster	Defined for each series identification
CAS program identification (CA_program_ID)	13	7.4.3.30	all	Provided and operated by broadcaster	
Data coding system identification*1 (data_component_id)	16	7.4.3.31 B.3	all	Provided by standardization organization	Registered and opened by application
Country region identification*1 (country_region_id)	6	7.4.3.32	all	Provided by standardization organization	Registered and opened after discussion
Component group type*1 (component_group_type)	3	7.4.3.33	all	Provided by standardization organization	Registered and opened after discussion
Component group identification*1 (component_group_id)	4	7.4.3.33	all	Provided by standardization organization	Registered and opened after discussion
Billing unit identification*1 (CA_unit_id)	4	7.4.3.33	all	Provided by standardization organization	Registered and opened after discussion

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Logo transmission type*1 (logo_transmission_type)	8	7.4.3.34	all	Provided by standardization organization	Registered and opened after discussion
Logo identification (logo_id)	9	7.4.3.34	all	Provided and operated by broadcaster	
Logo type (logo_type)	8	7.4.3.34	all	Provided by standardization organization	Registered and opened after discussion
Download identification (download_id)	32	7.4.3.36	all	Provided and operated by broadcaster	
Session identification (session_id)	32	7.4.3.37	all	Provided and operated by broadcaster	
Download protection system identification (DL_system_ID)	8	7.4.3.38	all	Provided by standardization organization	Registered and opened after discussion
Application format (application_format)	4	7.4.3.39	all	Provided by standardization organization	Registered and opened after discussion
EMT tag (EMT_tag)	8	7.4.3.39	all	Provided and operated by broadcaster	
Digital copy control information (digital_recording _control_data)	2	7.4.3.41	00, 10, 11	Provided by standardization organization	Registered and opened after discussion
			01	Provided and operated by broadcaster	
Broadcaster identification (ca_broadcaster_group _id)	8	7.4.3.45	all	Provided and operated by broadcaster	
Affiliation identification (affiliation_id)	8	7.4.3.46	all	Provided and operated by broadcaster	
Closed-caption identification tag (subtitle_tag)	8	9.2.2	all	Provided and operated by broadcaster	
Data type (data_type)	4	9.2.2	all	Provided by standardization organization	Registered and opened after discussion

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Closed-caption type (type)	2	9.3	all	Provided by standardization organization	Registered and opened after discussion
Closed-caption description system identification (subtitle_format)	4	9.3	all	Provided by standardization organization	Registered and opened after discussion
Compression type (compression_type)	4	9.3	all	Provided by standardization organization	Registered and opened after discussion
Compression identification (compression_type)	8	10.2.5	0x00 – 0xFE	Provided and operated by broadcaster	
			0xFF	Provided by standardization organization	Registered and opened after discussion
Data transmission session identification (data_transmission _session_id)	8	10.3.1.1	all	Provided and operated by broadcaster	
Application type (application_type)	16	10.3.2.1	all	Provided by standardization organization	Registered and opened after discussion
Node tag (node_tag)	16	10.3.2.2	all	Provided and operated by broadcaster	
Index item compression type (index_item_compression _type)	2	10.3.2.3	00, 01, 10	Provided and operated by broadcaster	
			11	Provided by standardization organization	Registered and opened after discussion
Index item identification (index_item_id)	32	10.3.2.3	all	Provided and operated by broadcaster	
Content identification (content_id)	16	10.3.2.4	all	Provided and operated by broadcaster	
Presentation unit tag (PU_tag)	8	10.3.2.4	all	Provided and operated by broadcaster	
Node tag (node_tag)	16	10.3.2.4	all	Provided and operated by broadcaster	

Identifier	Number of bits	STD-B60 provision section	range of value	prescriber	remarks
Transmission protocol label (transport_protocol _label)	8	10.3.3.1	all	Provided and operated by broadcaster	
Protocol identification (protocol_id)	16	10.3.3.2	all	Provided by standardization organization	Registered and opened after discussion
Data event identification (data_event_id)	4	11.2.1.1	all	Provided and operated by broadcaster	
Message group identification (event_msg_group_id)	12	11.2.1.1	all	Provided and operated by broadcaster	

*1: Identifier whose value is assigned from the same range as the provision in the ARIB STD-10

Description

Description 1 Relationship between MMT Package and Service

1 Service in Broadcasting Channel

The relationship between the MMT package and the service is shown in Fig. D1-1. “Service” as a continuation of programs which are transmitted on schedule is the same as the definition in the ARIB STD-B10. The unit of content is defined as a package in the MMT, and this package is used in one-to-one correspondence to the service. In the ARIB STD-B10, “program” of MPEG-2 Systems is used in one-to-one correspondence to the service, but in the MMT, the package is used instead of the program. The package corresponds to the service, and “program” which is delimited by start and end times in one service is the event.

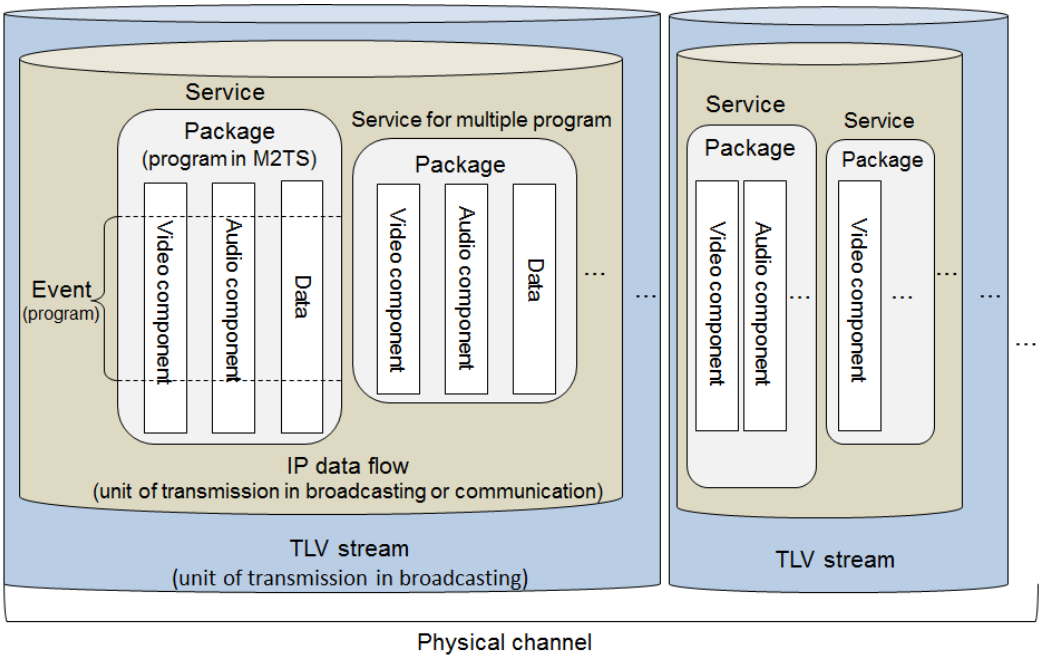


Fig. D1-1 The relationship between the MMT package and the service in broadcasting channel

In the MMT, components such as video and audio are defined as assets. Assets have a structure where MPUs are continuous.

The program is defined as a package which is composed of more than or equal to one asset and control message. The PA message is one of MMT-SIs, and MPT (MMT Package Table) which is involved in the PA message represents which asset composes the program.

As shown in Fig. D1-1, multiple MMT packages can be multiplexed in the same IP data flow. Here, the IP data flow is a set of IP packet which the values in the five kinds of fields; IP header and source IP address of UDP header, destination IP address, protocol type of IP header, source port number, and destination port number, are all the same. In addition to the IP dataflow which transmits the MMT package, an IP dataflow for download service and extension service may exist.

In broadcasting, such plural IP dataflows are multiplexed in one TLV stream. The TLV stream is a sequence of TLV packet which is identified by TLV stream ID, and involves the control information of TLV packet (TLV-SI) such as TLV-NIT and AMT. The transmission slot in which the TLV packet is multiplexed is specified from the TMCC signal in a transmission channel by

using the TLV stream ID.

2 Cross-Sectional Service of Broadcasting and Communication

In the MMT, the broadcasting transmission channel and the communication transmission channel can be handled in the same way. The configuration of the service which uses both broadcating transmission channel and communication transmission channel is shown in Fig. D1-2. Figure D1-2 shows that video component 1, audio component 1 and data 1 are transmitted by the broadcasting channel, and video component 2, audio component 2 and data 2 are transmitted by the communication channel. In broadcasting, three components to be transmitted are multiplexed in one IP data flow, and are transmitted by the same TLV stream. In the broadcasting channel, as the transmitted information is sent to all client terminal equipments, three components are multiplexed in one IP data flow. Also, as for the components to be transmitted by the communication channel, they are transmitted by independent IP data flows each other in order to comply with individual requests.

The MMT can easily realize hybrid distribution, as the components which are transmitted in the different transmission channels can be included in one package.

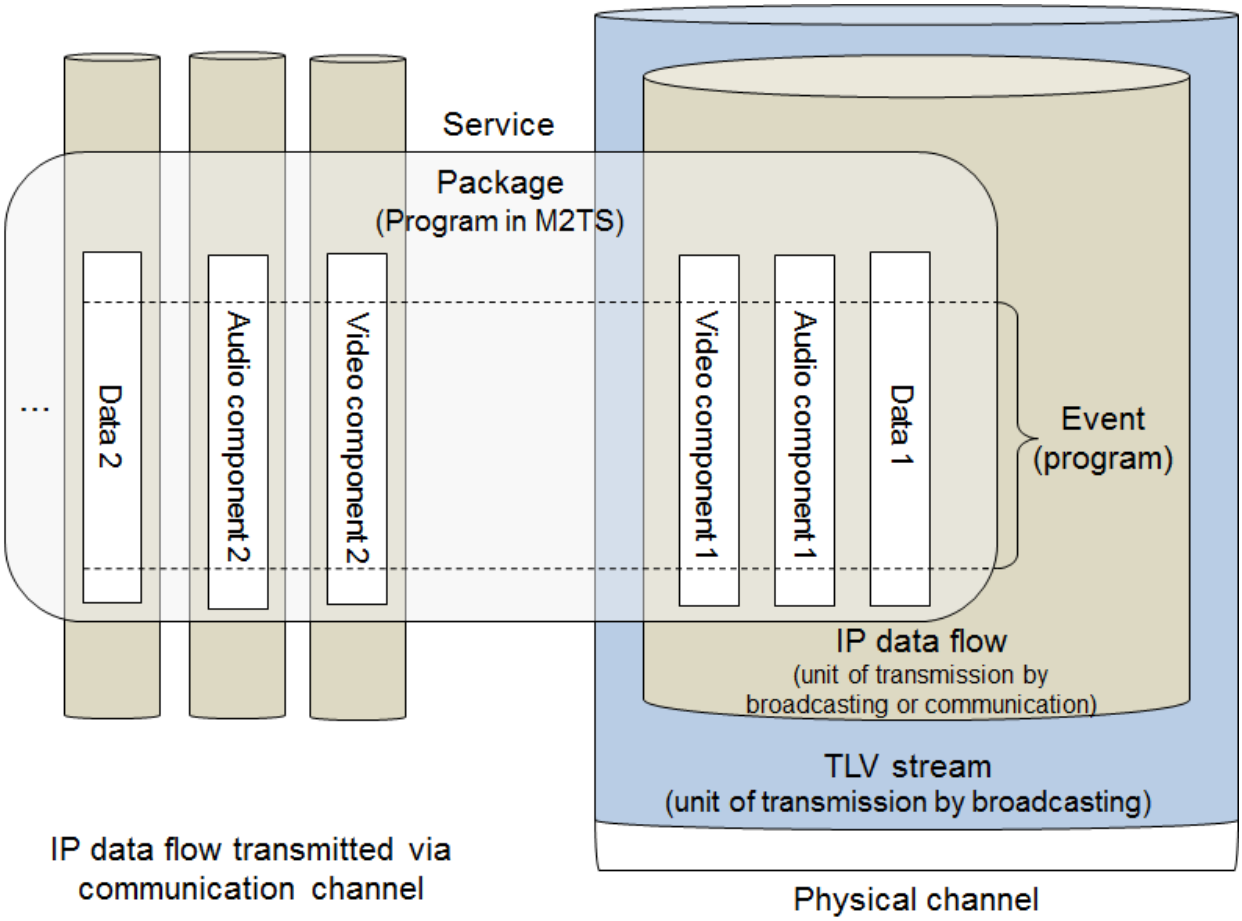


Fig. D1-2 Cross-sectional service of broadcasting and communication

Description 2　A Model of Receiving Buffer and Calculation Method of DTS/PTS

1　A Model of Receiving Buffer

A model of receiving buffer is shown in Fig. D2-1.

The receiver acquires IP packets from the payload of TLV packet which is received from the broadcasting channel, or directly acquires IP packets from the communication channel. The MMTP packet which is acquired by the payload of the received IP packets is stored in the MMTP transport buffer, and the receiving process of delivery layers such as absorption of transmission delay fluctuation, AL-FEC processing, etc. is performed. After that, it is classified to the MMTP packet of each asset such as video and audio based on the packet ID which is represented in MMTP packet header, and is stored in each MMTP buffer. From the stored payload of MMTP packet, NAL unit of video and audio, or LATM/LOAS stream format or data stream format is restored. These are input to the before decoding buffer, and output to video decoder or audio decoder at the timing of DTS, and decoded. Also, control messages are decoded to control informations by the system decoder, and various controls are performed.

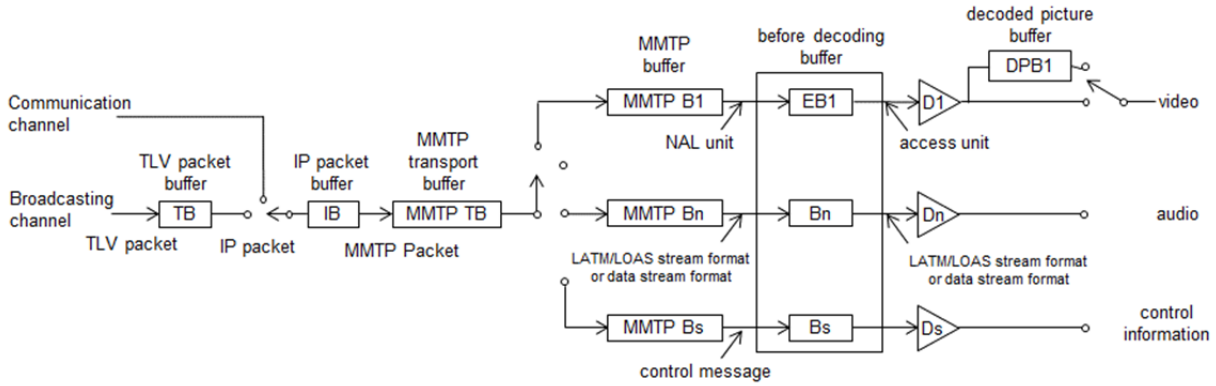


Fig. D2-1　A model of receiving buffer

Also, the before decoding buffer shown by EB1 in Fig. D2-1 is the CPB of video.

2　Calculation Method of DTS/PTS

The presentation times and decoding times of video and audio signals are supplied by using the MPU timestamp descriptor and the MPU extention timestamp descriptor. The summary of the access unit in the MPU, its presentation time and the decoding time is shown in D2-2.

The MPU timestamp descriptor supplies the presentation time of the first access unit in presentation order in the MPU by the format of NTP timestamp. It is the time indicated by mpu_presentation_time in Fig. D2-2.

The MPU extension timestamp descriptor supplies the differential value of decoding time of the first access unit in decoding order and the presentation time of the first access unit in presentation order (mpu_decoding_time_offset) in the MPU, the differential value (dts_pts_offest) of decoding and presentation times of each access unit, and the presentation period (pts_offset) of the access unit in the MPU by the value whose time scale is taken as time unit.

By using these time and values, the presentation and decoding times of n-th (n=1, 2, 3, 4, …) access unit in presentation order in the MPU can be calculated as follows.

Presentation time

$$\text{PTS (n)} = \text{mpu\_presentation\_time} + \sum_{k=1}^{n-1} \text{pts\_offset}(k) \div \text{timescale}$$

(Here, k =1, 2, 3, 4, … represents the order of presentation for access unit.)

Decoding time

$$\text{DTS (n)} = \text{PTS (n)} - \text{dts\_pts\_offset (n)} \div \text{timescale}$$

Also, the decoding time of the access unit which decodes first in the MPU is

$$\text{DTS}_{\text{initialAU}} = \text{mpu\_presentation\_time} - \text{mpu\_decoding\_time\_offset} \div \text{timescale}$$

The relationship is shown in Fig. D2-2.

Also, when the presentation period of the access unit is equal to the difference of the decoding time of the access unit and the decoding time of immediately after the access unit in decoding order, the decoding and presentation times of m-th (m=1, 2, 3, 4, …) access unit in decoding order in MPU can be calculated as the following.

Decoding time

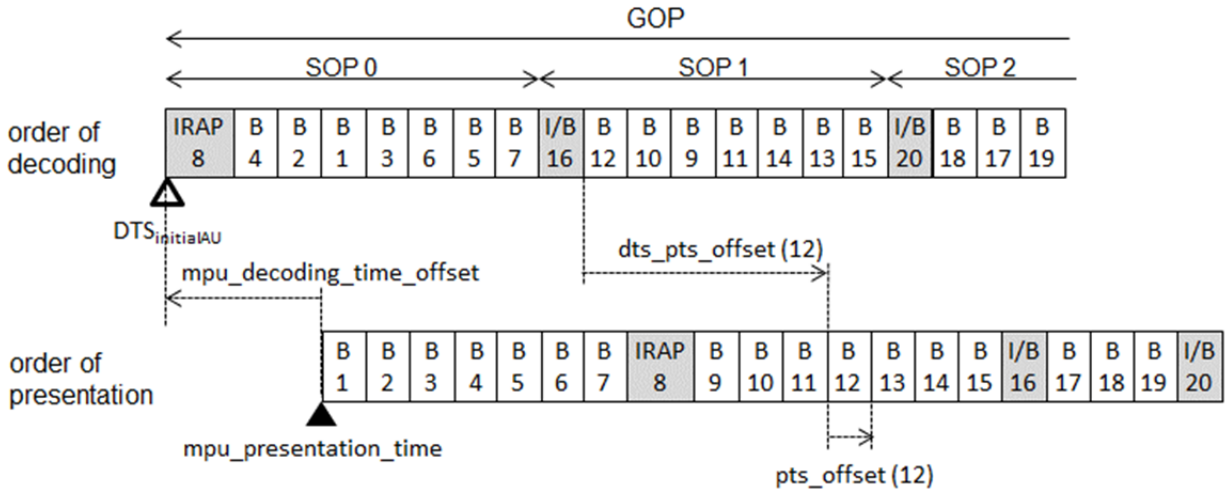
$$\text{DTS (m)} = \text{mpu\_presentation\_time} - \text{mpu\_decoding\_time\_offset} \div \text{timescale} + \sum_{l=1}^{m-1} \text{pts\_offset}(l) \div \text{timescale}$$

(Here, l =1, 2, 3, 4, … represents the order of decoding for access unit.)

Presentation time

$$\text{PTS (m)} = \text{DTS (m)} + \text{dts\_pts\_offset (m)} \div \text{timescale}$$

As calculated in this way, the MPU timestamp descriptor and the MPU extension timestamp descriptor are so transmitted that they can be received before the MPU starts decoding.



*The number in each access unit represents the order of presentation for the access unit.

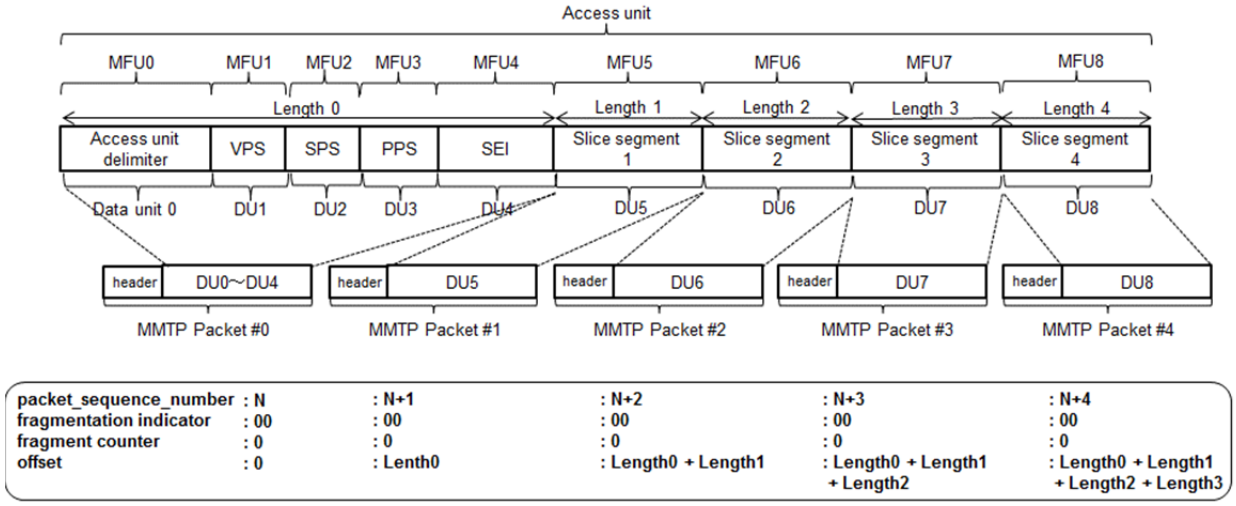
Fig. D2-2　Access unit in the MPU and its decoding and presentation times

Description 3　Decoding Method of Video Signal in the Receiver

1　Detection of Starting Point of Access Unit and Slice Segment

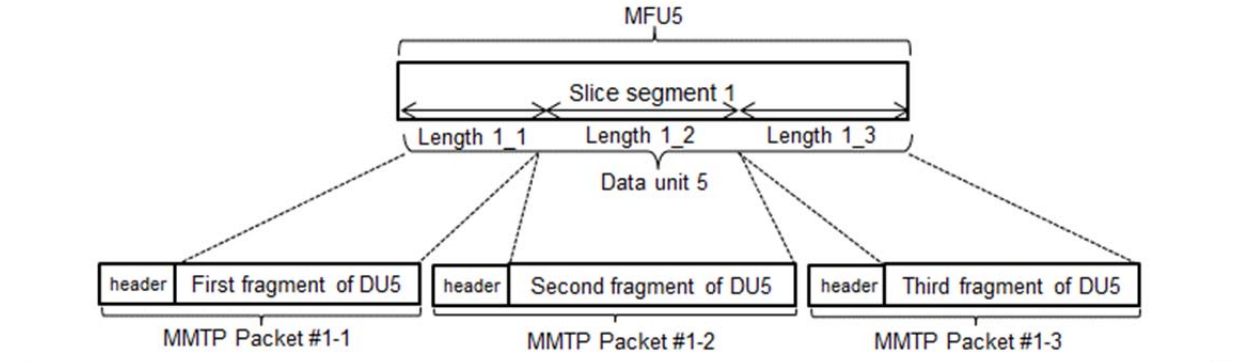
In this section, the method to specify the start point of the access unit and the slice segment based on the information of the transport layer without analyzing the video stream at the reciever will be explained.

Examples of the MMTP packet transmitting access unit, the MMTP packet header and the field values in the payload are shown in Fig. D3-1. Figure D3-1 (a) shows the case that the MTU is sufficiently big and the data unit is not divided, and Figure D3-1 (b) shows the case that the data unit is divided.



packet_sequence_number	: N	: N+1	: N+2	: N+3	: N+4
fragmentation indicator	: 00	: 00	: 00	: 00	: 00
fragment counter	: 0	: 0	: 0	: 0	: 0
offset	: 0	: Lenth0	: Length0 + Length1	: Length0 + Length1 + Length2	: Length0 + Length1 + Length2 + Length3

(a) In the case of not dividing data unit



packet_sequence_number	: N+1	: N+2	: N+3
fragmentation indicator	: 01	: 10	: 11
fragment counter	: 2	: 1	: 0
offset	: Length0	: (Length0 + Length1_1)	: Length0 + Length1_1 + Length1_2

(b) In the case of dividing data unit

Fig. D3-1　The MMTP packet that transmits the HEVC stream and an example of the field values of the MMTP payload

The sequence number of the MMTP packet (packet_sequence_number) with same packet ID is involved in the MMTP packet header. Also, for the MMTP payload, the presence/absence of the fragment of the data unit, the index number of the fragment in the data unit, or the information for specifying the start point of the fragment are represented in each field of fragmentation_indicator, fragment_counter and offset, respectively. By referring to these field values, the start points of the access unit and the slice segment can be specified.

The start point of the access unit is the NAL unit that is stored in the MMTP payload whose value of the offset field is 0. (By referring to sample_number field in the MMTP payload and judging the change of the index number of sample, the head of the access unit can be identified.) Also, the start point of the slice segment is the data unit that the value of the offset field is not '0' among the MMTP payloads whose fragment_indicator field is '00' or '01'.

As mentioned, the receiver can specify the start points of the access unit and the slice segment by using the information of the MMTP packet header and the MMTP payload.

2 Parallel Decoding Process of Video Signal

As 8K video signal is the heavy load for a single decoder to process decoding, it is also supposed that the access unit of the video signal is divided into four slice segments, and they are decoded by using four 4K decoders, as shown in Fig. D3-2.

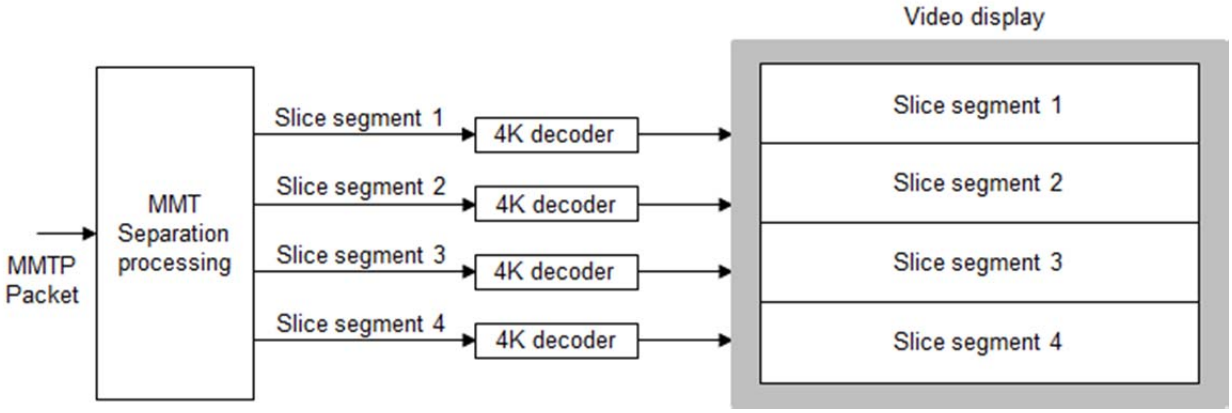


Fig. D3-2 Outline of implementation for parallel decoding by multiple decoders

The access unit is divided into the non-VCL NAL unit and the NAL unit of the slice segment, as shown in Fig. D3-3. After the non-VCL NAL unit and each slice segment are transformed to the connected configuration, they are input to each decoder. Then, specifying the start points of the access unit and the slice segment by using the method explained in the former section, the non-VCL NAL unit and the slice segment can be separated in the transport layer, and input data to each decoder can be generated.

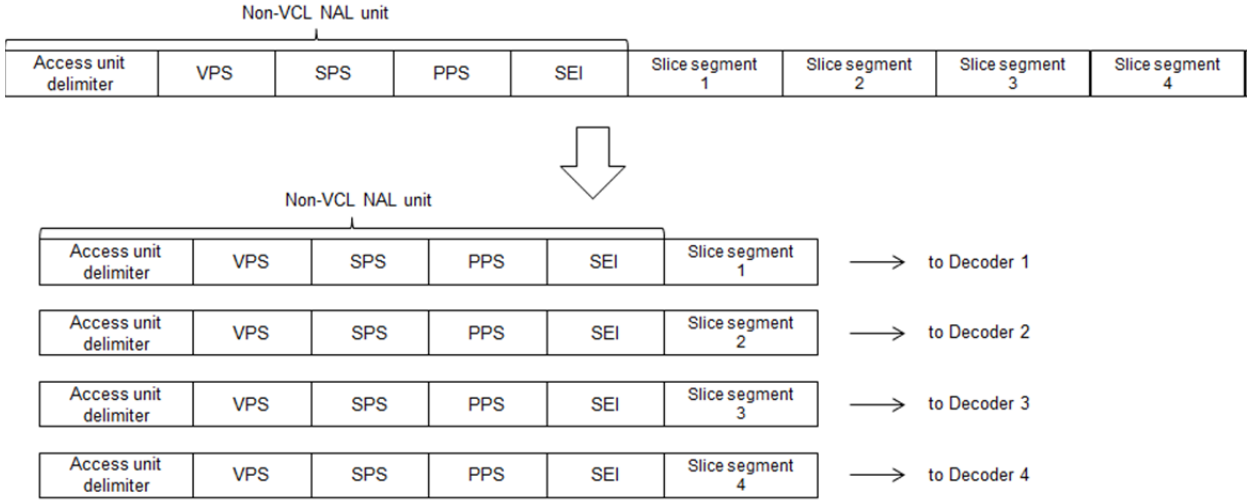


Fig. D3-3　Generation of data for parallel decoding

Description 4　An Example of Realization for NTP Clock Synchronization and Presentation Synchronization Based on VCO

Coordinated Universal Time is transmitted by the NTP timestamp format of 64 bits length, but it is actually difficult to realize the precision which expresses one second by 2^{32}. So, using as the base of the system clock for synchronizing the video system and as the system clock for operating the clock of the NTP format, it is considered to use not the 27 MHz which has been applied to the conventional broadcasting system, but the frequency of 2^n Hz. For example, the frequency of 2^{24} Hz≒16.8 MHz which is near to 27 MHz is supposed to apply, and the other frequencies of 2^{24} ～ 2^{28} Hz (268MHz), etc are supposed. The outline of a system model for the clock synchronization and the presentation synchronization in this example is shown in Fig. D4-1.

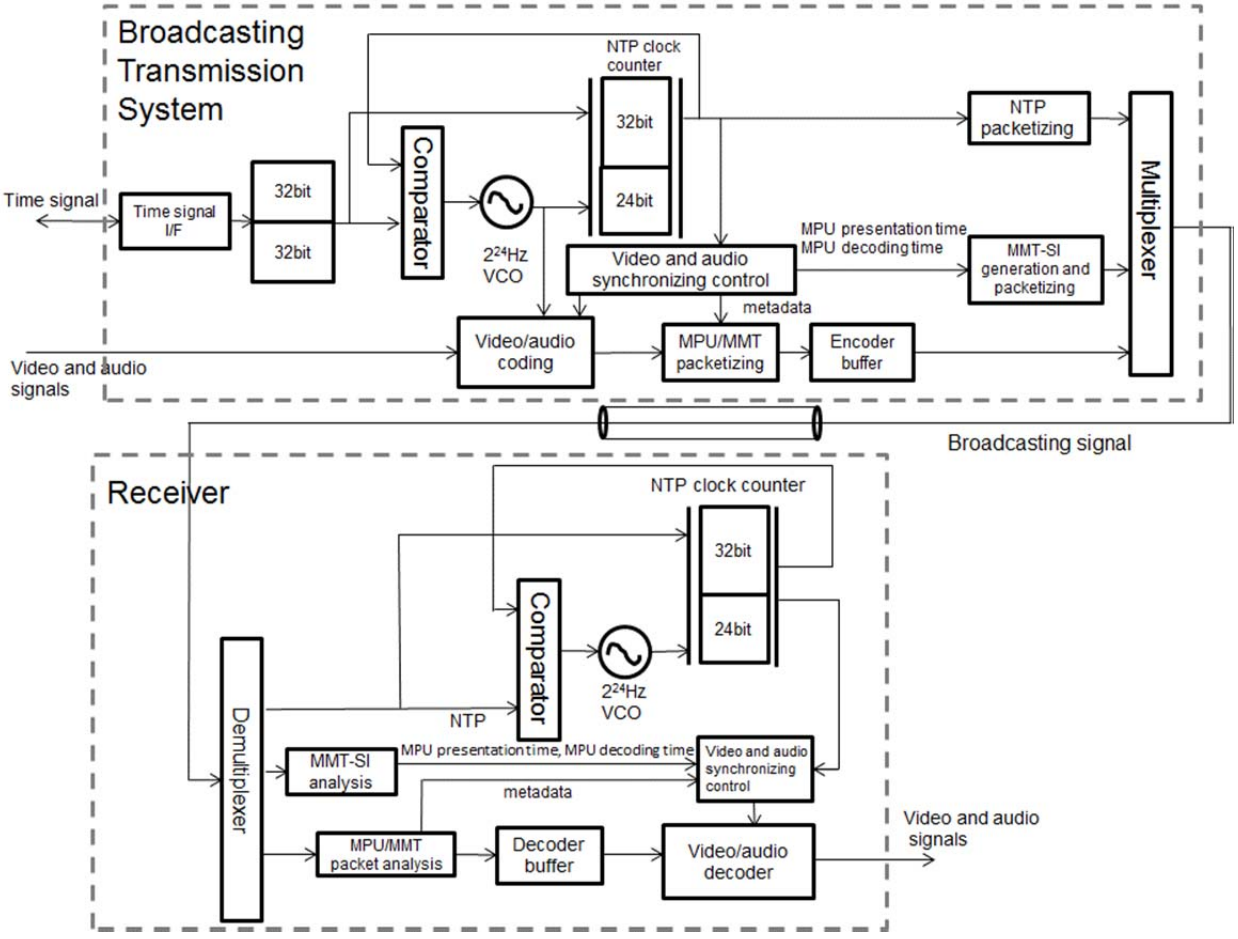


Fig. D4-1　System model for the clock synchronization and the presentation synchronization

In broadcast transmission system, by using IEEE 1588 (PTP) or NTP, configuring PLL by the counter made by 2^n Hz VCO after acquiring time information in the NTP format from the outside, the clock which synchronizes with time signal given from the outside is realized. Also, the whole signal processing system is made to work, synchronizing with the system clock of 2^n Hz. Moreover, the output of this clock is periodically transmitted in broadcasting channels as

the time information of NTP long format.

Decoding and presentation times per the presentation unit of video and audio signals are set based on the time information of the NTP format mentioned above. Therefore, it is necessary that metadata which is a part of MMT-SI and MPU is set and transmitted, considering this. Concretely, the time information of the MPU timestamp descriptor which is set by the unit of MPU is provided as the NTP long format, but it is necessary that the effective number of place is decided, considering the precision of the system clock and the required precision of the presentation synchronization. Whereas, the calculation unit of the time information of each presentation unit specified by the MPU metadata and the movie fragment metadata is not provided, but if setting to 2^n Hz equivalent to the effective number of place mentioned above, the quantity of operation in the receiver can be reduced. For example, in the case that 2^{16} Hz≒ 65536 Hz is applied, the precision is deteriorated a little bit worse than the precision of 90 kHz which has been applied to conventional broadcasting system, but the synchronization of video and audio signals is fully realized.

The receiver receives the time information of the NTP long format in broadcasting channel, and plays the receiving system clock by PLL based on 2^n Hz VCO as with the transmitting side. By this process, the receiving system clock becomes the clock which is synchronized with the broadcasting transmission system. Also, by working the signal processing system of the receiver in synchronizing with a system clock of 2^n Hz, the clock synchronization between the broadcasting transmission system and the receiver is realized, resulting in the stable signal reproduction.

In order to realize the presentation synchronization of video and audio signals in the receiver, the receiving system clock mentioned above, the calculated decoding time information in presentation unit in video and audio signals and the presentation time information are made to be matched.

Appendix

Appendix 1 Configuration of Control Information Described in This Standard

Among the control informations written in this standard, there are some with the same functions as the transmission control signals and the service informations provided in the ARIB STD-B10. So, in this appendix, about the control informations of MMT and TLV,

1) Those which are transferred from the STD-B10 (Table R1-1)

2) Those which are not in the STD-B10, but newly provided (Table R1-2)

will be shown.

And, those which are written in the STD-B10 and not transferred to the control informations of MMT and TLV are shown in Table R1-3.

Also, as for the tables and the descriptors provided in the ARIB STD-B10, while having the same functions, but those whose table identifiers and descriptor tag values are modified so as to be used in a broadcasting system using the MMT are discriminated from the original tables and descriptors by adding "MH-" which means MPEG-H to the head of their names.

Table R1-1 MMT and TLV control informations transferred from the STD-B10

	Table and Descriptor	Function
1	TLV-NIT	This transmits information which connects information of the transmission channel such as modulation frequency to broadcasting program in transmission by TLV packet.
2	AMT	This transmits information which connects service identifier identifying the broadcasting program number to IP packet.
3	Service list descriptor	This describes list of the program channel and its type.
4	Satellite delivery system descriptor	This describes the physical condition of satellite channel.
5	System management descriptor	This identifies broadcasting/non-broadcasting, etc.
6	Network name descriptor	This describes the network name.
7	ECM	This transmits program information (information about program and key for descramble, etc.) and common information composed of control information.
8	EMM	This transmits individual information involving contract information of each subscriber and work key for deciphering common information cipher, etc.
9	CA table (MH)	This transmits the descriptor about conditional access system.
10	MH-EIT	This transmits information about program such as program name, broadcasting date and time, explanation of content, etc.
11	MH-AIT	This transmits dynamic control information about application and additional information which is necessary for execution.
12	MH-BIT	This is used for presenting information of broadcaster which exists on network.
13	MH-SDTT	This transmits notice information such as service ID for download, schedule information, the kind of receivers which are the target

	Table and Descriptor	Function
		for updating, etc.
14	MH-SDT	This transmits information about program channel such as the name of program channel, the name of broadcaster, etc.
15	MH-TOT	This transmits the designation of present date and time and the differential time between actual time and display time for human system.
16	MH-CDT	This transmits data which is necessary for the receivers in common and is conditioned to be stored in non-volatile memory such as logo mark of broadcaster, etc.
17	Access control descriptor	This identifies conditional access system.
18	Emergency information descriptor (MH)	This supplies necessary information and description of function as emergency alarm signal.
19	MH-MPEG-4 audio descriptor	This describes basic information to specify coding parameters of MPEG-4 audio stream.
20	MH-MPEG-4 audio extension descriptor	This describes profile and lebel of MPEG-4 audio stream and proper setteing of the coding system.
21	MH-HEVC video descriptor	This describes basic coding parameters of video stream in ITU-T Recommendation H.265|ISO/IEC 23008-2 (HEVC stream).
22	MH-HEVC descriptor	This describes time information in ITU-T Recommendation H.265|ISO/IEC 23008-2 and HRD information.
23	MH-Linkage descriptor	This describes relationship with the other program channels.
24	MH-Event group descriptor	This describes grouping information of plural events.
25	MH-Service list descriptor	This describes the list of program channels and the kinds of them.
26	MH-Short event descriptor	This describes the name of program and simple explanation of the program.
27	MH-Extended event descriptor	This describes detailed information about program.
28	MH-Stream identifier descriptor	This is used for identifying individual signal of program element.
29	MH-Content descriptor	This describes the genre of program.
30	MH-Parental rating descriptor	This describes an age limit of permission for viewing.
31	MH-Audio component descriptor	This describes parameters about audio signal among program elements.
32	MH-Target region descriptor	This describes the region for target.
33	MH-Series descriptor	This describes series information extending to plural events.
34	MH-SI parameter descriptor	This describes parameters of SI transmission (group of period, period of re-transmission, etc.).
35	MH-Broadcaster name descriptor	This describes the name of broadcaster.
36	MH-Service descriptor	This describes the name of program channel and the name of the broadcaster.

	Table and Descriptor	Function
37	MH-Data component descriptor	This is used for identifying data coding system.
38	MH-Local time offset descriptor	When summer time is carried out, this describes the difference time between actual time (UTC+9 hours) and display time for human system.
39	MH-Component group descriptor	This describes grouping information of plural components.
40	MH-Logo transmission descriptor	This describes character string for simple logo, pointing to logo by CDT format,etc.
41	MH-CA startup descriptor	This describes information about startup CAS program which has the function of conditional access.
42	MH-Type descriptor	This represents the type of file which is transmitted by application transmission system.
43	MH-Info descriptor	This describes information about item.
44	MH-Expire descriptor	This describes the period of validity for item.
45	MH-Compression Type descriptor	This represents compression algorithm of item to be compressed and transmitted and the number of bytes of the item before compression.
46	MH-Hierarchy descriptor	This describes information to identify video stream component which is layer coded.
47	MH-CA contract information descriptor	This describes information to confirm that service or event can be reserved.
48	MH-CA service descriptor	This represents program channel of broadcaster who operates automatic display message, and this describes display control information of the concerned message.

Table R1-2 MMT and TLV control nformations
not described in the STD-B10, but newly provided

	Table and Descriptor	Function
1	PA message	This transmits table of MMT-SI as the entry point of MMT-SI.
2	M2 section message	This transmits section extension format of MPEG-2 Systems.
3	CA message	This transmits information about conditional access system.
4	M2 short section message	This transmits section short format of MPEG-2 Systems.
5	MMT package table	This gives information which configures package such as a list of asset and its position, etc.
6	Package list table	This represents a list of IP data flow and packet ID which transmits PA message of MMT package which is supplied as broadcasting service, and IP data folw which transmits IP service.
7	Layout configuration table	This is used for connecting layout information for presentation to the layout number.
8	DCM	This transmits key related information which consists of key, etc. to decode channel cipher for download.
9	DMM	This transmits key related information which consists of download key, etc. to decipher DCM.
10	Data directory management table	This supplies directory configuration of file which configures application.
11	Data asset management table	This supplies configuration of MPU in asset and the version information of each MPU.
12	Data content management table	This supplies configuration information of file as data content.
13	Event message table	This is used for transmitting information about event message.
14	Asset group descriptor	This supplies group relationship of asset and priority in group.
15	Event package descriptor	This supplies correspondence between event representing program and package.
16	Background color descriptor	This specifies the background color in layout designation.
17	MPU presentation region descriptor	This supplies the presentation position of MPU.
18	MPU timestamp descriptor	This supplies the presentation time of MPU.
19	Dependency descriptor	This supplies the asset ID of asset which is dependent.
20	Scramble descriptor	This identifies scramble sub-system.
21	Message authentification method descriptor	This identifies message authentification system.
22	Video component descriptor	This describes parameters, explanation, etc. about video signal among signals of program element.
23	IP data flow descriptor	This describes information of IP data flow involved in service.
24	UTC-NPT reference descriptor	This transmits relationship between NPT and UTC.

	Table and Descriptor	Function
25	Event message descriptor	This transmits information about event message in general.
26	MPU extention timestamp descriptor	This supplies decoding time, etc. of access unit in MPU.
27	MPU download content descriptor	This describes the attribute information of content which is downloaded by using MPU.
28	MH-Network download content descriptor	This describes attribute information of content which is downloaded by using network.
29	MH-Application descriptor	This describes information of application.
30	MH-Transport protocol descriptor	This describes the designation of transport protocol and location information of application which depends upon transport protocol.
31	MH-Simple application location descriptor	This describes details of acquisition destination of application.
32	MH-Application boundary and permission descriptor	This describes setting of application boundary, and broadcasting resource access authority per region (URL).
33	MH-Autostart priority descriptor	This describes startup priority of application.
34	MH-Cache control information descriptor	This describes information of cache control which caches and holds resource configuring application.
35	MH-Randomized latency descriptor	This describes the delay time for setting by which the timing to control application is probabilistically delayed.
36	Linked PU descriptor	This describes information of linked presentation unit.
37	Locked cache descriptor	This describes designation of the file of target for caching and locking.
38	Unlocked cache descriptor	This describes designation of the file for unlocking.
39	MH-Download protection descriptor	This describes location information and transmission information of MMTP packet which transmits DCM and DMM.
40	Application service descriptor	This describes entry information, etc. of application which is related to service.
41	MPU node descriptor	This represents the concerned MPU corresponds to directory node which is specified by data directory management table.
42	PU configuration descriptor	This represents list of MPU configuring presentation unit.
43	Remote control key descriptor	This sets service to assign to the selection button with one touch for remote controller of the receiver.
44	Content copy control descriptor	This describes information to control copy generation in the digital recording equipments and the maximum transmission rate.
45	Content use control descriptor	This describes information to control records and outputs.
46	Emergency news descriptor	This represents emergency news flash involving in security and safety (emergency earthquake news, news flash, prompt superimposition) is on the air.
47	Related broadcaster descriptor	This represents relationship to broadcaster of the other network in order to share NVRAM.
48	Multimedia service information descriptor	This describes detailed information about individual content of multimedia service.

Table R1-3 Those described in the STD-B10 and not transferred to the MMT and TLV control informations

	Table and Descriptor	Function	Table where descriptor is arranged	Reason for being not transferred
1	PMT			Because this is replaced by MP table.
2	PAT			Because this is replaced by package list table.
3	LDT (Linked Description Table)	This transmits information which reference information from the other table is collected to.		Because this is not transmitted when EIT [schedule] is operated.
4	TDT (Time and Date Table)	Designation of present date and time		Because this is not transmitted for BS/terrestrial digital broadcasting. Because TOT is used.
5	BAT (Bouquet Association Table)	Designation of information about bouquet (set of program channels), such as name of bouquet, involved program channel, etc.		Because this is not operated for BS/terrestrial digital broadcasting.
6	RST (Running Status Table)	Designation of present status of program		Because this is not transmitted for BS/terrestrial digital broadcasting.
7	LIT (Local Event Information Table)	Designation of information about local event such as identification (time) of local event (scene, etc.) in the program, name, explanation, etc.		Because this is not operated for BS/terrestrial digital broadcasting.
8	ERT (Event Relation Table)	This represents the relationship between programs and local events each other such as group, attribute, etc. of program and local event.		Because this is not operated for BS/terrestrial digital broadcasting.
9	ITT (Index Transmission Table)	Description of information about program index at the time for transmitting program		Because this is not operated for BS/terrestrial digital broadcasting.

	Table and Descriptor	Function	Table where descriptor is arranged	Reason for being not transferred
10	PCAT (Partial Content Announcement Table)	Notice of delivery of difference of content in data broadcasting		Because this is not operated for BS/terrestrial digital broadcasting.
11	ST (Stuffing Table)	Invalidation of table		Because this is dummy data.
12	NBIT (Network Board Information Table)	This transmits bulletin board information and the reference information for acquiring the bulletin board information.		Because bulletin board information is not used. (This is not transmitted in digital terrestrial television broadcasting.)
13	INT (IP/MAC Notification Table)	This transmits information which connects broadcasting program to IP/MAC stream which configures it in IP packet transmission by MPEG-2 TS.		Because of table for solving IP address for IP over TS.
14	DCT (Download Control Table)	Transmission of various information for separating and extracting DLT. This is provided in STD-B16.		Because this is not used for BS digital broadcasting.
15	DLT (Download Table)	Transmission of software for downloading. This is provided in STD-B16.		Because this is not used for BS digital broadcasting.
16	DIT (Discontinuity Information Table)	This indicates changing point where service information which is transmitted by partial transport stream may become discontinuity.		Because this is necessary when partial TS is output, and it is unnecessary in the IP circumstances.
17	SIT (Selection Information Table)	This indicates information about program which is transmitted by partial transport stream.		Because this is necessary when partial TS is output, and it is unnecessary in the IP circumstances.
18	Conditional access system descriptor	Description of conditional access	CAT, PMT	Because there is access control descriptor.

	Table and Descriptor	Function	Table where descriptor is arranged	Reason for being not transferred
		system and PID which transmits its ECM and EMM.		
19	Scalable transmission descriptor	Description of relationship between layered streams in scalable transmission	PMT	Because there is asset group descriptor.
20	TS information descriptor	This describes information about TS such as assignment of butten number of remote controller for the concerned TS and transmission layer of service in TS, etc.	NIT	Because “remote cotroller key descriptor” to be arranged in TLV-NIT is newly provided.
21	Stuff descriptor	Secure of space for descriptor, invalidation of descriptor	NIT, BAT, SDT, EIT, NBIT, LDT	Because this is dummy data.
22	Component descriptor	Description of type, explanation, etc. about program component signal.	PMT, EIT	Because it is replaced by video component descriptor.
23	Hyper-link descriptor	Description of link for the other program, the inside of program and information related program.	EIT, BIT	Because this is not used for BS digital broadcasting.
24	Video decode control descriptor	This is used for controlling video code at changing point of event.	PMT	Because EoS is added.
25	Reference descriptor	Description of node reference from program and local event.	EIT, LIT	Because this is not used for BS digital broadcasting.
26	Rectangle format node information descriptor	Name of node, and description of simple explanation	EIT, ERT	Because this is not used for BS digital broadcasting.
27	Extension broadcaster descriptor	This describes broadcaster information which is not limited in network.	BIT	Because this is not used for BS digital broadcasting.
28	AVC video descriptor	This describes profile and level of ITU-T Recommedation H.264|ISO/IEC	PMT	Because AVC is not provided.

	Table and Descriptor	Function	Table where descriptor is arranged	Reason for being not transferred
		14496-10 video.		
29	AVC timing HRD descriptor	This describes timing information for decoding ITU-T Recommendation H.264|ISO/IEC 14496-10 video.	PMT	Because AVC is not provided.
30	Service group descriptor	Description of grouping information of plural services	NIT	Because services are not connected.
31	Conditional playback system descriptor	Description of conditional playback system and PID which transmits its ECM and EMM.	CAT, PMT	Because conditional playback is not used.
32	Partial reception descriptor	Description of service identification which is transmitted in the partial receiving layer of terrestrial channel.	NIT	Because this is not used for BS digital broadcasting.
33	Terrestrial delivery system descriptor	Description of physical condition of terrestrial channel.	NIT	Because this is not used for BS digital broadcasting.
34	Intellectual property descriptor	Identification of intellectual property	PMT	Because this is not operated for BS/terrestrial digital broadcasting.
35	IP/MAC stream arrangement descriptor	Description of information about IP stream arrangement	INT	Because INT is not used.
36	Bouquet name descriptor	Description of bouquet name	BAT, SDT	Because this is not used for BS digital broadcasting.
37	Nation receivability descriptor	Description of object country for service.	PMT, BAT, SDT	Because this is not operated for BS/terrestrial digital broadcasting.
38	NVOD reference service descriptor	Description of the list of time shifted program channel for the reference program channel of Near VOD.	SDT	Because this is not used for BS digital broadcasting.
39	Time shift service descriptor	Description of the reference program channel for time shifted program channel of Near VOD.	EIT	Because this is not used for BS digital broadcasting.
40	Time shift event	Description of the	EIT	Because this is not

	Table and Descriptor	Function	Table where descriptor is arranged	Reason for being not transferred
	descriptor	reference program for time shifted program of Near VOD.		operated for BS/terrestrial digital broadcasting.
41	Mosaic descriptor	Description of the unit of division, connection to the other program channel and program about mosaic (divided pictures) service.	PMT, SDT	Because this is not operated for BS/terrestrial digital broadcasting.
42	CA identification descriptor	Description of conditional access system which can be used.	BAT, SDT, EIT	Because this is not operated for BS/terrestrial digital broadcasting.
43	Basic local event descriptor	Description of identification information of local event	LIT	Because this is not used for BS digital broadcasting.
44	Node relationship descriptor	Description of relationship from node to the other node	ERT	Because this is not used for BS digital broadcasting.
45	STC reference descriptor	Description of relationship with identification time of local event and STC	ITT	Because ITT is not operated.
46	SI prime TS descriptor	Description of identification information of SI prime TS and transmission parameter	BIT	Because this is not tranmitted for BS digital broadcasting except for inevitable occasion.
47	Bulletin board information descriptor	Description of the title of bulletin board and the text	NBIT	Because NBIT is not operated.
48	LDT link descriptor	After collecting descriptions which is referred from the other table, this is transmited.	EIT	Because LDT is not operated.
49	Connected transmission descriptor	Description of physical condition for connected transmission in terrestrial channel	NIT	Because this is not used for BS digital broadcasting.
50	Resistration descriptor	Description of information to identify private data which is not provided in ISO/IEC 13818-1.	PMT	Because of those which extends the stream type of MPEG-2 Systems.

	Table and Descriptor	Function	Table where descriptor is arranged	Reason for being not transferred
51	Data broadcast identification descriptor	Description of data broadcast identification	PMT	Because this is used only in INT.
52	Partial transport stream descriptor	Description about partial transport stream	SIT	Because this is not used for BS digital broadcasting.
53	Network identification descriptor	Description about network identification	SIT	Because this is not used for BS digital broadcasting.
54	Partial transport stream time descriptor	Description about partial transport stream time	SIT	Because this is not used for BS digital broadcasting.
55	Data content descriptor	Description of detailed information about each content of data program	EIT	Because this is extended in function as multimedia service information descriptor, and newly provided.
56	Caroursel compatible complex descriptor	A quasi-use of description function of descriptor defined in data carousel system		Because Data transmission JTG considered this, and it resulted in unnecessary because this is put on data asset management table.
57	Carousel identification descriptor	Description about carousel identification provided in ISO/IEC 13818-6		Because Data transmission JTG considered this, and it resulted in unnecessary for data transmission by MMT.
58	Association tag descriptor	Description about association tag information which is provided in ISO/IEC 13818-6		Because Data transmission JTG considered this, and it resulted in unnecessary for data transmission by MMT.
59	Extended association tag descriptor	Description about association tag information on the other broadcasting program which is provided in ISO/IEC 13818-6		Because Data transmission JTG considered this, and it resulted in unnecessary for data transmission by MMT.
60	CA_EMM_TS_ descriptor	When transmitting EMM by specific transponder system, this represents the specific transponder.		Because this is not used for BS digital broadcasting.

Appendix 2 Assignment Status of Identifier

1 Position of this Appendix

This appendix represents the assignment status of identifiers which the standardization organization provides, resisters and opens to the public.

1.1 Network Identification

The assignment status of network identification (network_id) is shown in Table R2-1. For the network identification, the value is assigned in the same range as the provision of the ARIB STD-B10. Refer to the STD-B10 Part 2, Annex N.

Table R2-1 Assignment status of network identification

Network identification	Name of network	Operational guidelines (reference)
0x000B	Advanced BS digital broadcasting	ARIB TR-B39
0x000C	Advanced wide band CS digital broadcasting	ARIB TR-B39

1.2 Data Coding System Identification

The assignment status of data coding system identification (data_component_id) is shown in Table R2-2. For the data coding system identification, the value is assigned in the same range as the provision of the ARIB STD-B10. Refer to the STD-B10 Part 2, Annex J.

Table R2-2 Assignment status of data coding system iIdentification

Data coding system identification	Meaning of data coding system
0x0020	Closed-caption coding system for digital broadcasting (2nd generation)
0x0021	Multimedia coding system for digital broadcasting (2nd generation)

1.3 Conditional Access System Identification

The assignment status of conditional access system identification (CA_system_id) is shown in Table R2-3. For the conditional access system identification, the value is assigned in the same range as the provision of the ARIB STD-B10. Refer to the STD-B10 Part 2, Annex M.

Table R2-3 Assignment status of data conditional access system identification

Conditional access system identification	Name of conditional access system	Operational guidelines (reference)
0x0005	ARIB conditional access system (2nd generation)	ARIB TR-B39

Attachment

Attachment 1 Guidelines on Operation of TLV-SI

1 Direction for Use of TLV-SI

1.1 Network Information Table for TLV (TLV-NIT)

1.1.1 Summary of Network Information Table for TLV (TLV-NIT)

Network Information Table for TLV (TLV-NIT) supplies the combination of TLV stream and the related tuning information. TLV-NIT can be used for initializing the receiver, and the related tuning information can be recorded in non-volatile memory. TLV-NIT can be also used for changing signal of tuning information. The following rules are applied to TLV-NIT.

--Transmission of TLV-NIT in the delivery system is essential.

--Only in the case that TLV-NIT involves the delivery system descriptor of the delivery system in being received, TLV-NIT which represents the delivery system in being received is valid.

This rule specifies the case that TLV-NIT involves an effective information. When transferring the boundary of broadcasting delivery system, TLV-NIT which is conveyed by the TLV stream can describe the previous network, but in this case, in order that the receiver acquires tuning information about the delivery system receiving, an different mechanism must be chosen. If a satellite receiver receives the satellite delivery system descriptor of the delivery system receiving, it is valid. However, if the receiver for cable receives the satellite delivery system descriptor of the delivery system receiving, it is of course invalid for the cable receiver.

--Valid TLV-NIT of the delivery system receiving must be able to have a look at all TLV streams in the delivery system receiving.

This rule makes the replacement of TLV-NIT at the boundary of the broadcasting delivery system easy. By simple replacement mechanism, the local frequency control is possible in a relatively cheap equipment.

For control information, two IDs; network identification (network_id) and original network identification (original_network_id) are used as labels related to the delivery system. The latter label plays a role to help to uniquely identify the service in the TLV stream, in the case that the TLV stream transferred to another delivery system which is not the original delivery system. Moreover making it clear, by following the root of original network identification / TLV stream identification / service identification, it is possible to uniquely refer to service. In this way, the network identification is not involved in this root. In the case that the service in a TLV stream transfers to another delivery system, only network identification changes and the original network identification does not change.

Figure G1-1 is an example of the case on assumption that two services (A and B) which are in two different delivery systems and their service identifications and TLV stream identifications are the same by chance, transfer to one new delivery system.

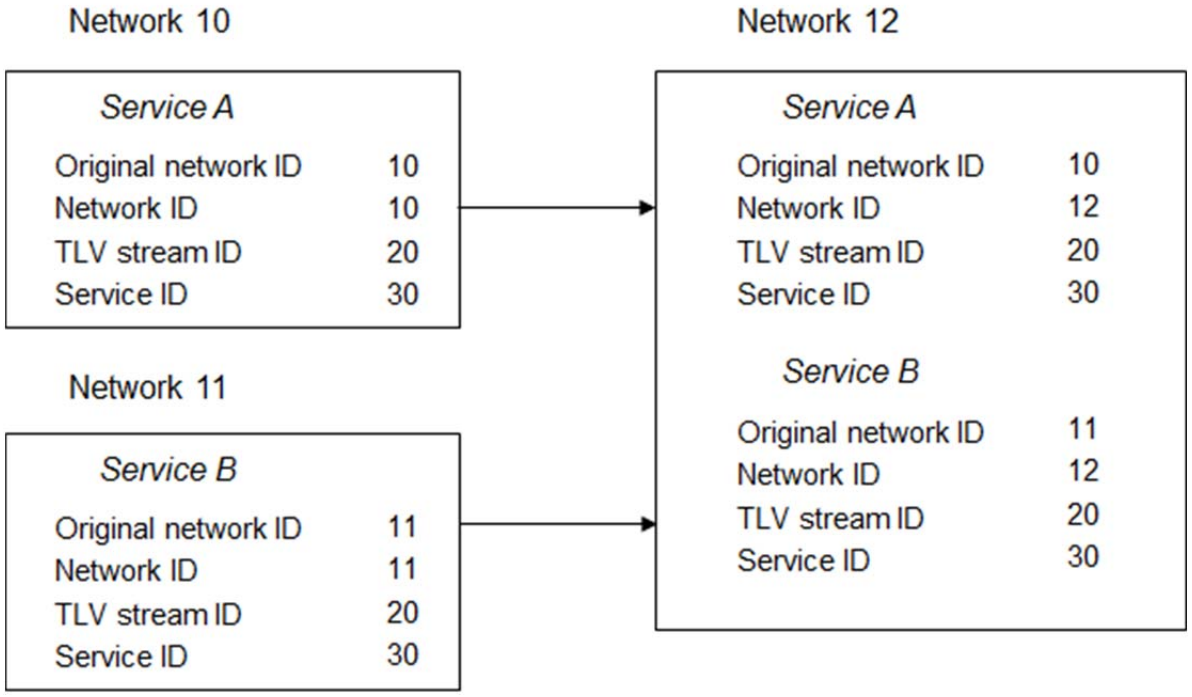


Fig. G1-1 Transfer to new delivery system

1.1.2 Descriptor of Network Information Table for TLV

1.1.2.1 Configuration of Descriptor of TLV-NIT

The configuration of TLV-NIT is shown in this standard, chapter 5.

1.1.2.2 First Descriptor Loop

For first descriptor loop of TLV-NIT, the descriptors which are defined in Ordinance and Notification are arranged.

1.1.2.3 Second Descriptor Loop

For second descriptor loop of TLV-NIT, besides the descriptors which are defined in Ordinance and Notification, descriptors in this section are arranged.

(1) Delivery system descriptor

The delivery system descriptor is used for transmitting physical parameters of delivery system which transmits the TLV stream.

The delivery system descriptor is arranged in each loop only once. In order to quickly tune to the TLV stream, the receiver needs to be able to interpret the delivery system descriptor.

(2) Service list descriptor

This descriptor supplies a list of service and kinds of service type in each TLV stream. Service is identified and listed by the service identification. The TLV stream identification and the original network identification are necessary to uniquely identify the service, and is presented at first in the descriptor loop.

The service list descriptor can be arranged in each loop only once. Transmission of this descriptor is optional in the other network. But it must be perfect in the case of transmitting.

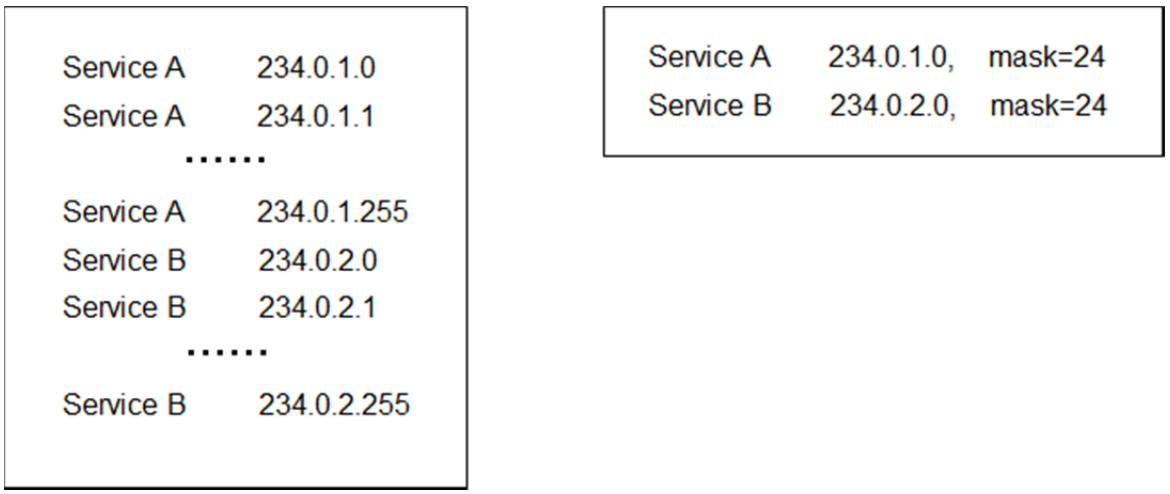
1.2 Address Map Table (AMT)

1.2.1 Summary of Address Map Table (AMT)

As an application receives the IP packet which is transmitted by TLV multiplex system and the IP packet which is transmitted via communication channel without distinguishing as possible, the Address Map Table (AMT) supplies a list of multicast group of the IP packet. The transmission of AMT by the TLV stream in being received is essential.

Besides ASM (Any Source Multicast) which specifies the multicast group only by group address, AMT also deals with SSM (Source Specific Multicast) which specifies the multicast group by combination of the source address and the group address as the mechanism of IP multicast.

Plural IP multicast group can be listed in one service identification. Address mask is used for efficiently describing successive IP addresses. By using the address mask, a list of plural IP addresses can be described in one line, as shown in Fig. G1-2. Also, by setting mask=32 for IPv4 and mask=128 for IPv6, the setting is equivalent to those which address mask is not used.



(a) In the cace of not using address mask (b) In the case of using address mask

Fig. G1-2 An example of efficient description of IP address by using address mask

1.2.2 Channel Selection by Multicast Group

By using AMT, channel selection becomes possible by designating IP multicast group. As an example of using AMT in the receiver, the example of tuning is shown in the case that external application of the receiver connects to the TLV multiplexing scheme combatible receiver via home network (Fig. G1-3).

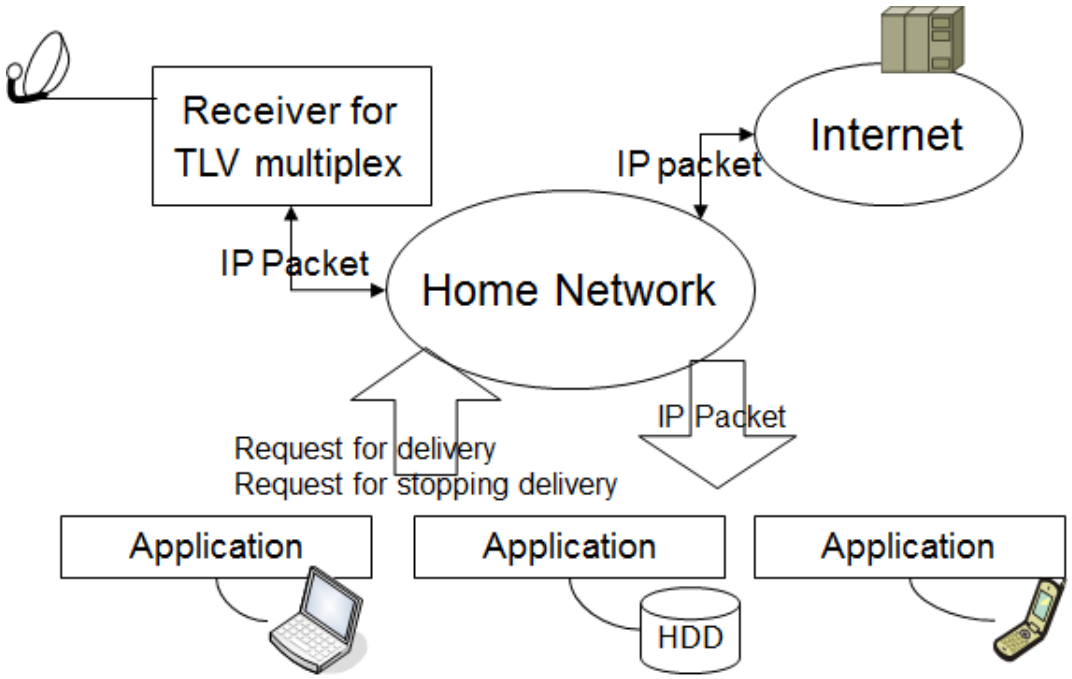


Fig. G1-3 The whole structure that application is connected to the TLV multiplexing scheme combatible receiver via home network

The functional block diagram of IP packet output part which is equipped in the TLV multiplexing scheme combatible receiver is shown in Fig. G1-4.

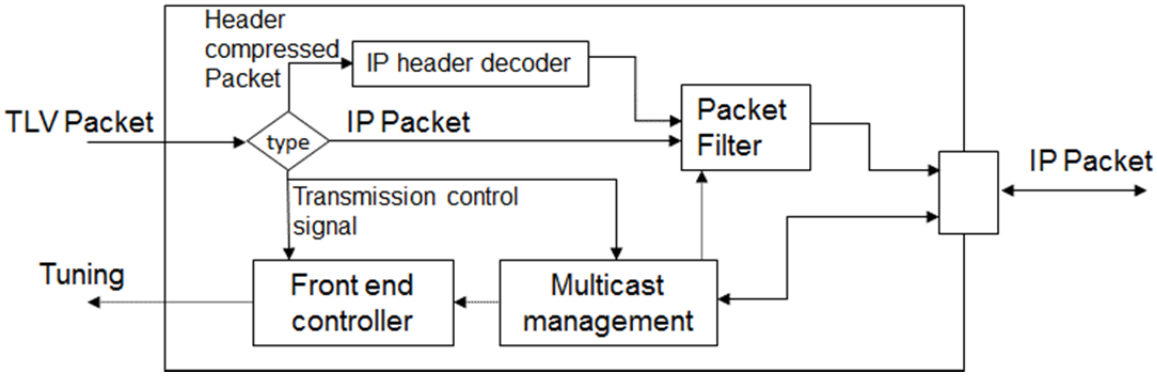


Fig. G1-4 Configuration of IP packet output part equipped in the TLV multiplexing scheme combatible receiver

For application's request for delivery or stopping delivery of IP packet to the TLV multiplexing scheme combatible receiver, IGMP or MLD which are the standard protocol used in IP multicast communication are used. Using join message at the start of receiving and using leave message at the end of receiving, the receiver which deals with TLV designates multicast group to be output to home network. The flowchart from application's delivery request to outputting designated IP packet from the receiver is illustrated as an example in Fig. G1-5. Also, the flowchart at the end of receiving is illustrated as an example in Fig. G1-6. These flowcharts are

shown as the case that application is one, but in the case that plural applications exist, each application may transmit the join message and the leave message.

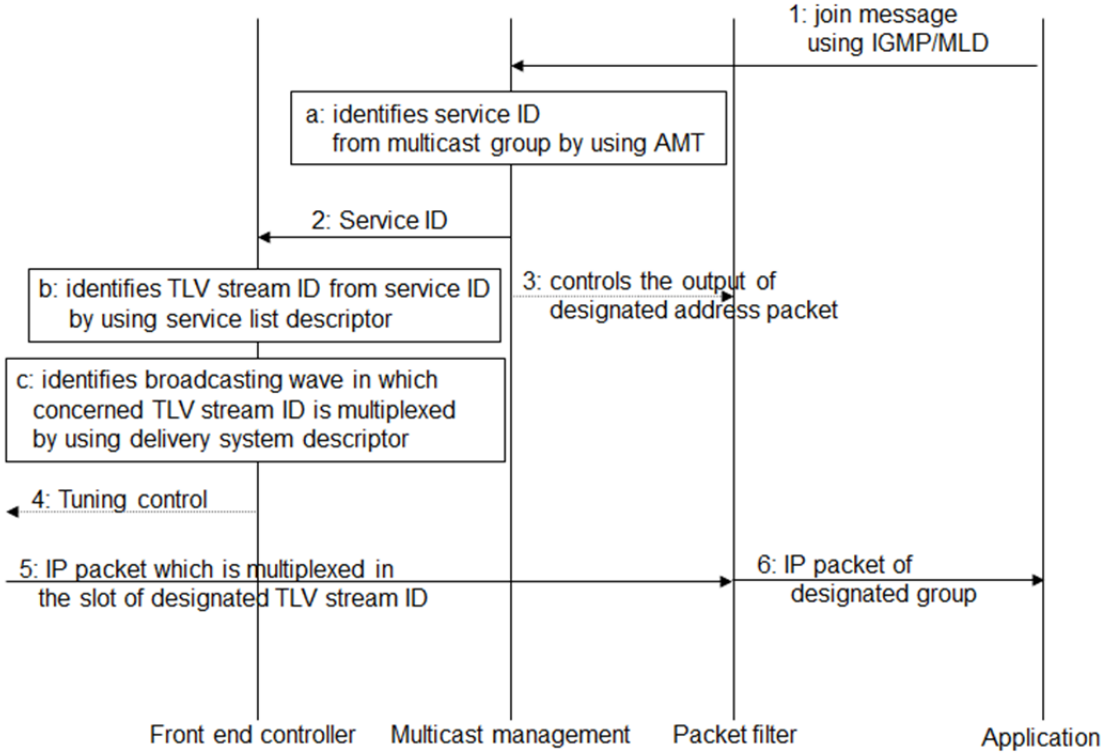


Fig. G1-5 An example of flowchart of the start of receiving IP packet by request for delivery

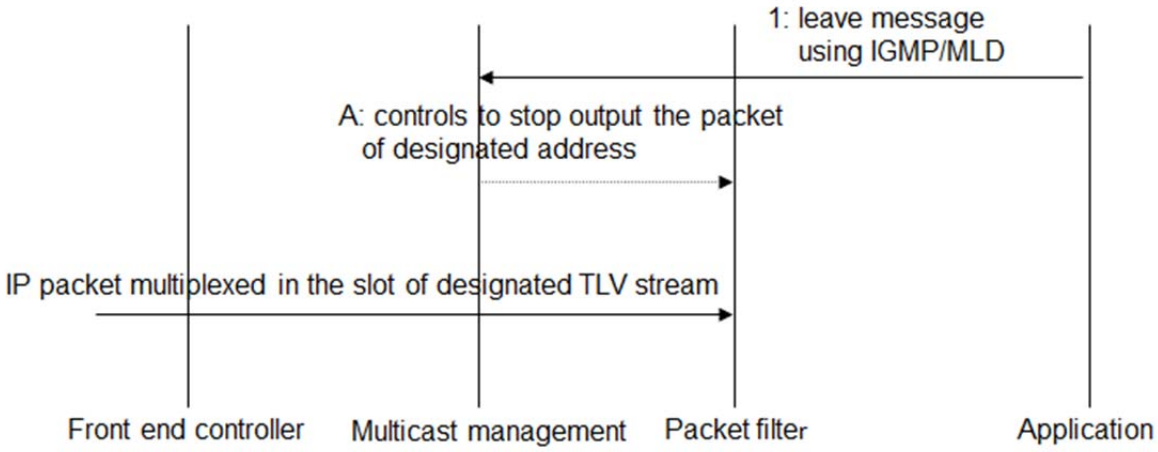


Fig. G1-6 An example of flowchart of the end of receiving IP packet by request for stopping delivery

Here, in the configuration which application is connected without going through home network, such that application is implemented in the same housing of the receiver, it is possible to tune even if the method such as IGMP and MLD is not used for request for delivery or request for stopping delivery.

(Note) IGMP: Internet Group Management Protocol, MLD: Multicast Listener Discovery

Protocol that the terminal use when it receives the delivery of IP multicast packet, and it requests the stop of delivery. There are different kinds according to the version of IP and the system of multicast.

IGMPv2 (RFC 2336): This is used for IPv4. It deals with Any Source Multicast.

IGMPv3 (RFC 3376): This is used for IPv4. It deals with Source Specific Multicast.

MLDv1 (RFC 3376): This is used for IPv6. It deals with Any Source Multicast.

MLDv2 (RFC 2710): This is used for IPv6. It deals with Source Specific Multicast.

IGMPv3 is upper compatible to IGMPv2, and MLDv2 is upper compatible to MLDv1.

MMT-BASED MEDIA TRANSPORT SCHEME IN DIGITAL BROADCASTING SYSTEMS

ARIB STANDARD

ARIB STD-B60 VERSION 1.13-E1
(October 11, 2018)

This Document is based on the ARIB standard of “MMT-BASED MEDIA TRANSPORT SCHEME IN DIGITAL BROADCASTING SYSTEMS” in Japanese edition and translated into English in October, 2018.

Published by

Association of Radio Industries and Businesses

11F, Nittochi Building
1-4-1 Kasumigaseki, Chiyoda-ku, Tokyo 100-0013, Japan

TEL 81-3-5510-8590
FAX 81-3-3592-1103

Printed in Japan