ENGLISH TRANSLATION

Data Coding and Transmission Specifications for Digital Broadcasting

ARIB STANDARD

ARIB STD-B24 Version 5.0

VOLUME 3

Established on	June	20,	2000	Version 1.0
Revised on	June	28,	2002	Version 3.2
Revised on	February	5,	2004	Version 4.0
Revised on	May	29,	2006	Version 5.0

Association of Radio Industries and Businesses (ARIB)

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

1. The copyright of this document is ascribed to the Association of Radio Industries and Businesses (ARIB).

2. 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.

3. The ARIB Standards and ARIB Technical Reports are usually written in Japanese and approved by the ARIB Standard Assembly. This document is a translation into English of the approved document for the purpose of convenience of users. If there are any discrepancies in the content, expressions, etc., between the Japanese original and this translated document, the Japanese original shall prevail.

4. The establishment, revision and abolishment of ARIB Standards and Technical Reports are approved at the ARIB Standard Assembly, which meets several times a year. Approved ARIB Standards and Technical Reports, in their original language, are made publicly available in hard copy, CDs or through web posting, generally in about one month after the date of approval. 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

Contents

Preface

Volume 1 Data Coding

Part 1 Reference Model for Data Broadcasting

Part 2 Monomedia Coding

Part 3 Coding of Caption and Superimpose

Volume 2 XML-based Multimedia Coding Scheme

Appendix 1 Operational Guidelines

Appendix 2 Operational Guidelines for Implementing Basic Services

Appendix 3 Operational Guidelines for Implementing Extended Services for Fixed Receiving System

Appendix 4 Operational Guidelines for Implementing Extended Services for Portable Receiving System

Appendix 5 Operational Guidelines for Implementing Extended Services for Mobile Receiving System

Volume 3 Data Transmission Specification

Preface

ARIB (Association of Radio Industries and Businesses) establishes the "ARIB Standards" for the basic technical conditions of standard specifications related to variety of radio communication equipments, broadcasting transmission equipments, and its reception equipments using radio wave with the participation of radio communication equipment manufacturers, broadcasting equipment manufacturers, electric communication companies, service providers and other users.

"ARIB Standards" are nongovernmental standards established by combining governmental technical standards established for the purpose of effective use of frequency and to avoid interference of other users, and nongovernmental optional standards established for convenience for radio communication equipment manufacturers, broadcasting equipment manufacturers, electric communication companies, service providers and users, in order to secure appropriate quality and compatibility of radio communication equipment and broadcast equipment, etc.

This standard is established for "Data Coding and Transmission Specification for Digital Broadcasting" by the approval of the standardization committee, participated by radio communication equipment manufacturers, broadcast equipment manufacturers, electric communication companies, service providers and users irrespectively, to secure impartiality and clearness.

For data broadcasting of digital broadcasting, it is directed by the Telecommunications Technology Council on July 21, 1999 that it is desired that the most desirable multimedia coding specification in Japan at this point should be based on an XML-based specification, which is superior in many points such as "function", "contents production environment", "compatibility with other media", "data processing at terminal side", "extension ability of coding method", and "future direction of engineering development", etc., and that the detailed specifications should be standardized by the nongovernmental standardization organization with flexibility.

This standard is established as nongovernmental standard of data broadcasting specification used in Japan based on this direction, and consists of three parts: mono-media coding, multimedia coding, and data transmission specification. Compatibility with multiplex data broadcasting specification, which is already used in Japan is considered for mono-media coding. Compatibility with network usage or data broadcasting method in Europe and America is considered for multimedia coding and the coding scheme is based on XML coding specified in W3C specification adding necessary specifications for broadcasting. Each coding scheme in this standard is applied to whole broadcasting media generally and the conditions proper to broadcasting media derived from transmission methods and service requirements should be specified as operational restrictions.

Though this standard is mainly applied to BS digital broadcasting as the first step, the specification should be completed adding necessary specifications for other broadcasting media, considering trends of international standardization and new technological trends which cannot be assumed yet.

We hope that this standard will be put to practical use actively by radio communication equipment manufacturers, broadcast equipment manufacturers, electric communication companies, service providers, users, and so on.

Notice:

This standard does not describe industrial proprietary rights mandatory to this standard. However, the right proprietor of the industrial proprietary rights has expressed that "Industrial proprietary rights related to this standard, listed in the annexed table below, are possessed by the applicator shown in the list. However, execution of the right listed in the annexed table below is permitted indiscriminately, without exclusion, under appropriate condition, to the user of this standard. In the case when the user of this standard possesses the mandatory industrial proprietary rights for all or part of the contents specified in this standard, and when he asserts his rights, it is not applied."

Annexed table

Patent applicant	Name of invention	Patent number	Remarks
Matsushita Electric Industrial Co., Ltd.	情報処理装置	特開平 04-205415号	Japan
	データサーバ装置及び端末装置	特開平 06-139173号	Japan
	放送を用いて対話性を実現する送信装置、受信装置、受信方法、その受信プログラムを記録した媒体、通信システム	特開平 10-070712号	Japan
	データ入出力端末装置	特開平 10-074134号	Japan
	情報処理装置	特開平 10-083270号	Japan
	ﾃﾞｰﾀの提示を制御するﾃﾞｰﾀ提示制御装置、ﾃﾞｰﾀの提示を～情報を送信するﾃﾞｰﾀ送信装置及びﾃﾞｰﾀ～ﾃﾞｰﾀ提示制御情報編集装置	特開平 10-164530号	Japan
	ﾃﾞｼﾞﾀﾙ放送ｼｽﾃﾑ、ﾃﾞｼﾞﾀﾙ放送装置及びﾃﾞｼﾞﾀﾙ放送における受信装置	特開平 10-304325号	Japan
	ﾃﾞｼﾞﾀﾙ放送装置、受信装置、ﾃﾞｼﾞﾀﾙ放送ｼｽﾃﾑ、受信装置に適用するﾌﾟﾛｸﾞﾗﾑ記録媒体	特開平 10-313449号	Japan
	番組編集装置および番組受信装置	特願平 10-020585号	Japan
	放送局ｼｽﾃﾑ及び受信機	特願平 10-195093号	Japan
	ﾃﾞｼﾞﾀﾙ放送のための記録再生装置および方法	特願平 11-367308号	Japan
	ﾃﾞｰﾀ送受信ｼｽﾃﾑおよびその方法	特願平 11-103619号	Japan
	ﾃﾞｼﾞﾀﾙﾃﾞｰﾀ送受信ｼｽﾃﾑおよびその方法	特願平 11-124986号	Japan
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver3.8 *5		
TOSHIBA CORPORATION	多重放送ｼｽﾃﾑとこのｼｽﾃﾑで使用される放送送信装置および放送受信装置	特開平 09-162821号	Japan

Patent applicant	Name of invention	Patent number	Remarks
	ﾃﾞｼﾞﾀﾙ放送装置及びﾃﾞｼﾞﾀﾙ放送方法、ﾃﾞｼﾞﾀﾙ放送受信装置及びﾃﾞｼﾞﾀﾙ放送受信方法、ﾃﾞｼﾞﾀﾙ放送受信ｼｽﾃﾑ*16	特許第3621682号	Japan
NHK (Japan Broadcasting Corporation)	文書情報出力装置および方法	特開平 9-244617号	Japan
	入力データの自動選択処理装置	特開平 11-328189号	Japan
	ﾏﾙﾁﾒﾃﾞｨｱ型情報ｻｰﾋﾞｽ方式およびその方式の実施に使用する装置	特開平 11-331104号	Japan
Sony Corporation *1	音声信号圧縮方法及びﾒﾓﾘ書き込み方法	特許第 1952835号	Japan
	オーディオ信号処理方法	特許第 3200886号	Japan
	オーディオ信号処理方法	特許第 3141853号	Japan
	信号符号化又は複合化装置、及び信号符号化又は複合化方法、並びに記録媒体	WO94/28633	Japan
	信号符号化方法及び装置、信号複合化方法及び装置、並びに記録媒体	特開平 7-168593	Japan
	符号化音声信号の複合化方法	特開平 8-63197	Japan
	音声信号の再生方法、再生装置及び伝送方法	特開平 9-6397	Japan
	音声信号の再生方法及び装置、並びに音声複合化方法及び装置、並びに音声合成方法及び装置、並びに携帯無線端末装置	特開平 9-190196	Japan
	音声符号化方法、音声複合化方法及び音声符号化複合化方法	特開平 8-69299	Japan
	音声符号化方法及び装置、音声複合化方法及び装置	特開平 9-127991	Japan
	符号化データ複合化方法及び符号化データ複合化装置	特許 2874745号	Japan
	映像信号符号化方法	特許 2877225号	Japan
	符号化データ編集方法及び符号化データ編集装置	特許 2969782号	Japan
	動画像データエンコード方法及び装置、並びに動画像データデコード方法および装置	特許 2977104号	Japan
	動きベクトル伝送方法及びその装置並びに動きベクトル複合化方法及びその装置	特許 2712645号	Japan
Mitsubishi Electric Corporation	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver3.1 *2		
	マルチメディア多重方式*3	特許第 3027815号	Japan

Patent applicant	Name of invention	Patent number	Remarks
	マルチメディア多重方式*3	特許第 3027816号	Japan
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.4 *15		
Motorola Japan Ltd.	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver3.6 *4		
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver3.8 *5		
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver3.9 *6		
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.0 *7		
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.1 *9		
NTT DoCoMo, Inc. *11	動画像符号化方法、動画像複合方法、動画像符号化装置、及び動画像複合装置*11	特許第 3504256号	Japan, EPC, USA, Korea, China, Taiwan
	動画像符号化方法、動画像複合方法、動画像符号化装置、動画像複合装置、動画像符号化プログラム、及び動画像複合プログラム*11	特許第 3513148号	Japan, EPC, USA, Korea, China, Taiwan
	動画像複合方法、動画像複合装置、及び動画像複合プログラム*11	特許第 3534742号	Japan, EPC, USA, Korea, China, Taiwan
	信号符号化方法、信号複合方法、信号符号化装置、信号複合装置、信号符号化プログラム、及び、信号複合プログラム*11	特許第 3491001号	Japan, EPC, USA, Korea, China, Taiwan
	インターリーブを行うための方法および装置並びにデ・インターリーブを行うための方法および装置*13	特許第 3362051号	Japan, USA, Korea, Singapore, Australia, China

Patent applicant	Name of invention	Patent number	Remarks
	誤り保護方法および誤り保護装置*13	特許第 3457335号	Japan, USA, UK Korea, Germany, France Italy, Singapore, Australia, China
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.4 *15		
Sharp Corporation *5	画像符号化装置および画像復号装置	特許第 2951861号	Japan
NEC Corporation *5	画像信号の動き補償フレーム間予測符号化・複合化方法とその装置	特許第 1890887号	Japan
	圧縮記録画像の再生方式	特許第 2119938号	Japan
	圧縮記録画像の対話型再生方式	特許第 2134585号	Japan
	適応変換符号化の方法及び装置	特許第 2778128号	Japan
	符号化方式および復号方式	特許第 2820096号	Japan
	変換符号化複合化方法及び装置	特許第 3070057号	Japan
	改良DCTの順変換計算装置および逆変換計算装置	特許第 3185214号	Japan
	適応変換符号化方式および適応変換複合方式	特許第 3255022号	Japan
Philips Japan, Ltd	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.0 *8		
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.1 *10		
	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.2 *12		
Philips Electronics Japan, Ltd.	Submitted comprehensive confirmation of patents applied to the revised parts of ARIB STD-B24 Ver4.3 *14		

Note) *1: valid for the revised parts of ARIB STD-B24 Ver3.0
*2: valid for the revised parts of ARIB STD-B24 Ver3.1
*3: valid for the revised parts of ARIB STD-B24 Ver3.3
*4: valid for the revised parts of ARIB STD-B24 Ver3.6
*5: valid for the revised parts of ARIB STD-B24 Ver3.8
*6: valid for the revised parts of ARIB STD-B24 Ver3.9 （accepted on October 9,2003)
*7: valid for the revised parts of ARIB STD-B24 Ver4.0 （accepted on January 8,2004)
*8: valid for the revised parts of ARIB STD-B24 Ver4.0 （accepted on January 29,2004)
*9: valid for the revised parts of ARIB STD-B24 Ver4.1 （accepted on November 17,2004)
*10: valid for the revised parts of ARIB STD-B24 Ver4.1 （accepted on December 7,2004)

*11: valid for the revised parts of ARIB STD-B24 Ver3.8（accepted on January 7,2005)

*12: valid for the revised parts of ARIB STD-B24 Ver4.2（accepted on March 14,2005)

*13: valid for the ARIB STD-B24 Ver1.0（accepted on September 26,2005)

*14: valid for the revised parts of ARIB STD-B24 Ver4.3（accepted on September 27,2005)

*15: valid for the revised parts of ARIB STD-B24 Ver4.4（accepted on March 6,2006)

*16: valid for the revised parts of ARIB STD-B24 Ver3.6（accepted on March 14,2006)

VOLUME 3

Data Transmission Specification

[BLANK]

Contents

Chapter 1 Purpose 1

Chapter 2 Scope 2

Chapter 3 Definitions and Abbreviations 3

3.1 Definitions 3

3.2 Terminology 3

3.3 Abbreviations 6

Chapter 4 Types of data transmission specification 8

Chapter 5 Independent PES transmission specification 9

5.1 Synchronized PES 9

5.2 Asynchronous PES 10

Chapter 6 Data carousel transmission specification 11

6.1 Transmission with DSM-CC data carousel 11

6.2 Download Info Indication (DII) message 11

6.2.1 Syntax and semantics of DII message 11

6.2.2 Syntax and semantics of dsmccMessageHeader() 13

6.2.3 Descriptors of module information area and private area 14

6.3 DownloadDataBlock (DDB) message 24

6.3.1 Syntax and semantics of DDB message 24

6.3.2 Syntax and semantics of dsmccDownloadDataHeader() 25

6.4 Syntax of dsmccAdaptationHeader() 26

6.5 Syntax of DSM-CC section 26

Chapter 7 Event message transmission specification 29

7.1 Stream descriptor 29

7.1.1 NPT reference descriptor 29

7.1.2 General event descriptor 30

7.2 Syntax of DSM-CC section transmitting stream descriptor 32

Chapter 8 Interaction Channel Protocols for Digital Broadcasting 34

8.1 Data Transfer Protocols 34

8.1.1 Line connection and disconnection Phase 34

8.1.2 DataLink establishment and termination phase 34

8.1.3 Protocols Used to Direct Connections (Data Transmission Phase) 35

8.1.4 Protocols for Mass Calls Reception Service 39

8.1.5 Protocols for interaction channel carrying request and broadcast channel carrying response 39

8.2 Security 39

8.2.1 Data encryption 40

8.2.2 Data integrity.. 40
8.2.3 Authentication of communicating party.. 41
8.2.4 Signature 42

Annex A Video PES... 47

Annex B Audio PES... 48
B.1 Data transmission format of Audio PES coded with MPEG-2 BC Audio 48
B.2 Data transmission format of Audio PES coded MPEG-2 AAC Audio 48

Annex C PSI/SI information for data carousel transmission and event message transmission........... 49
C.1 Content of additional_data_component_info loop of data_component_descriptor................ 49
C.2 Selector byte of data_contents_descriptor... 49
C.2.1 Data structure for data carousel reception control for non-stored data services........... 49
C.2.2 Data structure for data carousel reception control for stored data services.................. 51

Informative Explanation.. 54
1 Supplementary Notes on PES Transmission.. 54
2 Supplementary Notes on Data Carousel Transmission 54
3 Relationship among Local Content, Content, and Data Content Descriptors 54
4 Applications of StoreRoot/Subdirectory descriptor for pre-loaded data services...................... 55
5 Connection Models for Interactive Systems 56
6 Cryptographic Methods for Interactive Channel Transmission 60

Reference materials 64

Chapter 1 Purpose

This standard specifies data transmission protocol for data broadcasting, which is carried out as part of the digital broadcasting that is specified as Japanese standard.

Chapter 2 Scope

The standard described in Volume 3 is applied to data transmission for data broadcasting carried out as part of digital broadcasting.

Chapter 3 Definitions and Abbreviations

3.1 Definitions

content: A group of data transmitted through a data broadcasting program or a bidirectional communications service used together with the data broadcasting program to serve as part of the data broadcasting program. More specifically as a term in the broadcasting context, the term " content" indicates a set of streams in the program sending the group of data. A single data broadcasting program can be composed of multiple contents.

data event: A set of data broadcasting streams that represents a group of data broadcast content to be distributed with the start time and end time preconfigured. The concept of "data event" is introduced in order to allow a group of data broadcasting content to be switched to another whether or not they are in the same program, as needed. In other words, a data event is independent of an event.

local content: A piece of data broadcasting content contained in a single data event. Generally, a local content is a group of data clustered based on the context or for a better convenience of program production.

reserved [undefined]: The term "reserved", when used in the clauses defining the coded bit stream, indicates that the value may be used in the future for ISO defined extensions. Unless otherwise specified within this standard, all reserved bits must be set to 1.

reserved_future_use [undefined]: The term " **reserved_future_use** ", when used in the clauses defining the coded bit stream, indicates that the value may be used for extensions defined by this standard in the future. Unless otherwise specified within this standard, all reserved bits must be set to 1.

3.2 Terminology

audio frame: A minimum unit which can be decoded into an individual set of audio signals in the structure of audio data specified in ISO/IEC13818-3.

data carousel: A method that sends out any set of data repeatedly so that the data can be downloaded via broadcasting in any timing as needed. This method is specified in ISO/IEC13818-6.

DSM-CC U-U: The User-to-User interface specified in ISO/IEC 13818-6.

DSM-CC U-N: The User-to-Network download protocol specified in ISO/IEC 13818-6.

PES packet: A data format used to transmit elementary streams. A PES packet consists of a PES packet header and a PES payload immediately following the header.

PES packet header: A field that comprises the first part of a PES packet.

PES stream: A contiguous stream of PES packets containing the same elementary stream. Each PES stream has a unique stream_id.

picture header: A part that describes attributions of a picture layer in a video data structure , as specified in ISO/IEC 13818-2.

private_stream_1: A type of a stream transmitted using PES as specified in ISO/IEC 13818-1. This stream is used to transmit a private stream that needs to be synchronized with other streams.

private_stream_2: A type of a stream transmitted using PES as specified in ISO/IEC 13818-1. This stream is used to transmit a private stream that does not need to be synchronized with other streams.

section: A syntactic structure used to map service information and other data into a transport stream packet, as specified in ISO/IEC 13818-1.

transport buffer: A buffer for receiving a transport stream packet in a target decoder of a MPEG2 transport stream, as specified in ISO/IEC13818-1.

ADSL: (Asymmetric Digital Subscriber Line) A technology for data transmission using existing twisted-pair lines for phone services, in which upstream and downstream data speed are different.

adaptation format: A format used in a DSM-CC header. An adaptation format is an information format inserted in an adaptation field that encodes information to meet a request depending on the delivery network.

ARP: (Address Resolution Protocol) A protocol used in a TCP/IP network to obtain an Ethernet node's physical address based on its IP address.

ASN.1: (Abstract Syntax Notation.1) A general encoding method used to deliver structured data. Each data object is described as a composition of an identifier, a value representing the length of the object's content, and the content itself, arranged in this order. The presence of the length information enables binary representation of the content.

BASIC mode data transmission protocol: A communication protocol developed for basic data transmission between a host and a terminal. The protocol employs a method for minimizing transmission errors.

CBC mode: A cipher algorithm supported by a shared key cryptosystem. This mode employs an IV (Initialization Vector) value that is a result of an XOR operation whose operands are an encoded value and an input value preceded by the encoded value.

CDMA Cellular System: (Code Division Multiple Access Cellular System) A cellular phone system based on the CDMA scheme. Data can be transmitted at 9600 bps and 14400 bps over a switched circuit network and at 144 kbps over a packet switching network.

center: A facility equipped with a host necessary to provide bidirectional transmission service.

CHAP: (Challenge Handshake Authentication Protocol [RFC 1994]) A PPP (Point-to-Point Protocol) component that handles authentication. This protocol encryptes a user ID and a password upon sending to ensure a higher level of security than that implemented by the PAP (Password Authentication Protocol).

Code independent mode: An extended method of BASIC mode data transmission protocol which can transmit binary data.

Collection/distribution network: A network that collects and distributes data from and to receivers in a widespread deployment.

Congestion: A state where a telephone network cannot establish requested connections because the requested connections overload the network in terms of throughput. A congestion may be worse when failed connection requests are repeated until they are established.

DHCP: (Dynamic Host Configuration Protocol [RFC 1541]) A protocol used to automatically configure terminals on a TCP/IP network. For example, this protocol allows IP addresses to be assigned dynamically.

DNS: (Domain Name Service [RFC 1034, RFC 1035]) A protocol used by the service that maps a host name on a network into its IP address.

DS-CDMA: (Direct Spread Code Division Multiple Access) A CDMA (Code Division Multiple Access) scheme that employs direct spreading.

Ethernet: ([IEEE 802]) A LAN standard that defines a bus-based network employing the CSMA/CD (Carrier Sense Multiple Access/Collision Detection) for access control.

FTP: (file transfer protocol [RFC 959]) A protocol used to transfer and share a file between two hosts communicated via TCP/IP.

FTTH: (Fiber To The Home) An infrastructure of a high-speed, broadband-based network that provides optical fiber connections to every house and integrates separate services including telephone services, accesses to the Internet, and television broadcasting services, etc.

Hash function: A mathematical function that maps a large (a very huge in some cases) area into a small range. It is necessary for a good hash function to be both one-way and collision-free.

HDLC procedure: (High-level Data Link Control) A transmission control procedure with high reliability used for communication among computers mainly on LANs and the Internet.

Host: An access point device or a server device necessary for bidirectional transmission services.

HTTP: (HyperText Transfer Protocol [RFC 1954]) An application layer protocol used to transmit data for World Wide Web.

ICMP: (Internet Control Message Protocol [FRC 792]) A protocol used to send a message that occurs during data transfer. These messages include various error notifications and validation prompts.

IMT-2000: (International Mobile Telecommunication 2000) A mobile communications scheme that supports both wired-quality voice connections with cellular phones and high-speed data transmissions over mobile devices at up to 2 Mbps.

IP: (Internet Protocol [RFC 791]) A network layer protocol that defines the addressing mechanism on the Internet to allow data to be transmitted.

IPCP: (IP Control Protocol [RFC 1332]) A protocol used to establish various configurations required to use IP on the network layer protocol phase.

LCP: (Link Control Protocol) A PPP (Point-to-Point Protocol) component that establishes data-link connections.

MC-CDMA: (Multi Carrier Code Division Multiple Access) A CDMA (Code Division Multiple Access) scheme that employs multi carrier systems.

MIME: An application layer protocol that provides a content architecture that allows multimedia data such as non-US-ASCII format text files, audio files, and image files to be transmitted via Internet e-mail.

MNP4: An error correction method for modem communication.

NNTP: (Network News Transfer Protocol [RFC 977]) An application layer protocol used to distribute, post and retrieve Net News on the Internet.

PAP: (Password Authentication Protocol [RFC 1334]) A PPP (Point-to-Point Protocol) component that supports authentication. This protocol does not encrypt a user ID and a password upon sending.

PDC: (Personal Digital Cellular) A switched circuit method for communication via digital automobile phones and cellular telephones. It is also capable of data communication at 9600 bps.

PDC-P: (Personal Digital Cellular-Packet) A packet switching method for communication via digital automobile phones and cellular telephones. It is capable of data communication at 9600 bps or 28.8 kbps.

PIAFS: (PHS Internet Access Forum Standard) A data communication protocol used to transmit data at up to 32 kbps or 64 kbps via the PHS (the personal handyphone system, a digital mobile telephone system based on a Japanese standard).

POP3: (Post Office Protocol version 3 [RFC 1939]) A protocol used to retrieve and delete an e-mail or a list of e-mails from a spool in a mail server.

PPP: (Point to Point Protocol [RFC 1661]) A protocol designed to transfer multiple protocols via a point-to-point linkage. PPP is used for dial up connections.

PPP in HDLC-like Flaming: A flame structure to serve as a higher level protocol than PPP. This is a method to configure a header and a footer based on a frame structure used by a HDLC protocol.

PPP Internet Protocol Control Protocol Extensions for name Server Addresses: ([RFC 1877]) A protocol used to resolve a name server using the PPP.

PPPoE: (PPP over Ethernet [RFC 2516]) A protocol that enables PPP frames to be transmitted over an Ethernet network.

SMTP: (Simple Mail Transfer Protocol [RFC821]) A protocol used to relay and deliver e-mails.

TCP: (Transmission Control Protocol [RFC 793]) A transport layer protocol that provides highly reliable end-to-end, connection-oriented data delivery using an error detection and correction mechanism.

Telnet: [RFC 854, RFC 855] A protocol used to provide a virtual terminal that allows a terminal to access a remote server on a TCP/IP network.

TLS: (Transport Layer Security [RFC 2246]) A protocol used to send and receive encrypted data over the Internet. This protocol supports a combination of various security technologies including PKC, SKC, digital certificates, and hash functions, to prevent eavesdropping, message forgery, and spoofing.

TTC JT-I-430: A specification that applies to physical interfaces related to the Layer 1 indicated by the Telecommunication Technology Committee in Japan.

TTC JT-Q.931: A specification that applies to configuration, maintenance, connection breakage/recovery on a net connection related to the Layer 3 indicated by the Telecommunication Technology Committee in Japan.

TTC JT-Q.921: A specification that applies to elements of a procedure and formats of a field related to link access procedure operation regarding the Layer 2 indicated by the Telecommunication Technology Committee in Japan.

UDP: (User Datagram Protocol [RFC 768]) A transport layer protocol that provides connectionless data delivery between two hosts. UDP does not support acknowledgement messages, and it minimizes protocol overhead for higher transmission efficiency services.

V.22bis: An ITU-T Recommendation that defines full-duplex modem communication over PSTN up to 2400 bps.

V.32bis: An ITU-T Recommendation that defines full-duplex modem communication over PSTN up to 14.4 kbps.

V.34: An ITU-T Recommendation that defines full-duplex modem communication over PSTN up to 33.6 kbps.

V.42bis: An ITU-T Recommendation that defines a data compression and error correction method for modem communication.

X.28: A scheme used to convert communication protocols in order to connect a non-packet communication device to a packet switching network.

3.3 Abbreviations

ATSC: (Advanced Television System Committee) A committee for the purpose of digital broadcasting system standardization in the USA.

DAVIC: (Digital Audio Visual Council) An industrial consortium for interactive multimedia service standardization.

DVB: (Digital Video Broadcasting) A project for digital broadcasting system standardization in Europe.

DMS-CC: (Digital Storage Media Command and Control) A control method defined in ISO/IEC 13818-6. This method supports accessing files and streams in digital interactive service.

EIT: (Event Information Table) An event information table which contains data related to an event and a program such as an event (program) name, a start time and a duration.

SI: (Service Information) Digital data that describes an arrangement of programs, a delivery system for broadcasting data streams, descriptions of programs, schedule/timing information, etc. This data also conveys MPEG-2 PSI (Program Specific Information) and extension parts defined independently.

PID: (Packet Identifier) A packet identifier of an MPEG-2 Transport Stream.

PSI: (Program Specific Information) A transmission control information specified in ISO/IEC 13818-1, which provides information required to allow a receiver to automatically demultiplex and decode various program streams that have been multiplexed.

PMT: (Program Map Table) A table that is a part of the PSI. This table indicates a location (PID of transport stream packet) of a program map table corresponding to each service in the multiplexed stream.

ISDN: (Integrated Services Digital Network) Integrated Services Digital Network

PSTN: (Public Switched Telephone Network) Public Switched Telephone Network

3.4 Terminology Used in Ministerial Ordinances and Notifications

As the identifiers used in this standard to identify Service Information and other data comply with the notation rule specified in ARIB STD-B10, any relationship between an identifier and a Ministerial Ordinance and Notification should follow those specified in ARIB STD-B10.

Chapter 4 Types of data transmission protocol

The types of the data transmission protocols and the stream types contained in a PMT, as specified in this standard, are shown in Table 4-1.

Table 4-1 Types of transmission protocol

Transmission protocol	Major functions and usages	Stream type
Independent PES transmission protocol	Used for streaming synchronous and asynchronous data for broadcasting services. Applied to subtitles and superimposed characters.	0x06
Data carousel transmission protocol	Used to transfer general synchronous and asynchronous data for broadcasting services. Applied to data transmission for download services and multimedia services.	0x0B, 0x0D*
Event message transmission protocol	Used for synchronous and asynchronous message notification to an application on the receiver unit from the broadcasting station. Used in multimedia services.	0x0C, 0x0D**
Interaction Channel protocols	Transmission protocols used in a fixed network such as a PSTN/ISDN network and a mobile network including a mobile phone/PHS network when bidirectional communication is also used in a broadcasting service.	-

* When a stream contains no DSM-CC data other than a data carousel, 0x0B or 0x0D is used and when it also has other DSM-CC data, 0x0D is used.

** When a stream contains no DSM-CC data other than an event message, 0x0C or 0x0D is used and when it also has other DSM-CC data, 0x0D is used.

Chapter 5 Independent PES transmission protocol

The independent PES transmission protocol is a method used to implement streaming for data broadcasting services. The independent PES transmission protocol defined in this chapter has two types: synchronized type and asynchronous type.

The synchronized PES transmission protocol is used when it is necessary to synchronize data in a stream with other streams including video and audio. The asynchronous PES transmission protocol is used when the synchronization is not necessary. As a major application example, it is expected that the synchronized type is used for transmitting captions and the asynchronous type is used for transmitting superimposed characters.

5.1 Synchronized PES

According to the synchronized PES transmission protocol, data is transmitted using a PES packet specified in ISO/IEC 13818-1. Any mapping of a PES packet to an MPEG-2 transport stream must comply with ISO/IEC 13818-1.

A PES packet with the following restrictions is used in addition to the syntax and semantics specified in ISO/IEC 13818-1.

- The PES packet header corresponding to the private_stream_1 is used.
- stream_id: In the case of a synchronized type stream, it is set to ‘0xBD‘(private_stream_1).
- PES_packet_length: This 16-bit field has a non-zero value.
- The synchronized PES data structure shown in Table 5-1 is inserted into the PES_packet_data_bytes field.

Table 5-1 Synchronized PES data structure

Syntax	Bits	Mnemonic
Synchronized_PES_data() {		
data_identifier	8	uimsbf
private_stream_id	8	uimsbf
reserved_future_use	4	bslbf
PES_data_packet_header_length	4	uimsbf
for (i=0; i<N1; i++) {		
PES_data_private_data_byte	8	bslbf
}		
for (i=0; i<N2; i++) {		
Synchronized_PES_data_byte	8	bslbf
}		
}		

Semantics of fields in a Synchronized PES packet:

data_identifier: This 8-bit field is set to ’0x80’.[1]

private_stream_id: Unused (0xFF)

PES_data_packet_header_length: This 4-bit field indicates the length in bytes of the PES_data_private_date_bytes.

PES_data_private_data_byte: This is an 8-bit field and a more detailed usage of this field depends on a service. A receiver unit may skip this field.

Synchronized_PES_data_byte: This is an 8-bit field containing the transmitted data.

[1] Refer to Informative Explanation 1 (1).

5.2 Asynchronous PES

According to the asynchronous PES transmission protocol, data is transmitted using a PES packet specified in ISO/IEC 13818-1. Any mapping a PES packet to an MPEG-2 transport stream must comply with ISO/IEC 13818-1.

A PES packet with the following restrictions is used in addition to the syntax and semantics specified in ISO/IEC 13818-1.

- The PES packet header corresponding to private_stream_2 is used.
- stream_id: In case of an asynchronous type stream, it is set to ‘0xBF'(private_stream_2).
- PES_packet_length: This 16-bit field has a non-zero value.
- The asynchronous PES data structure shown in Table 5-2 is inserted to the field of the PES_packet_data_bytes.

Table 5-2 Asynchronous PES data structure

Syntax	Bits	Mnemonic
Asynchronous_PES_data() {		
data_identifier	8	uimsbf
private_stream_id	8	uimsbf
reserved_future_use	4	bslbf
PES_data_packet_header_length	4	uimsbf
for (i=0; i<N1; i++) {		
PES_data_private_data_byte	8	bslbf
}		
for (i=0; i<N2; i++) {		
Asynchronous_PES_data_byte	8	bslbf
}		
}		

Semantics of fields in an Asynchronous PES packet:

data_identifier: This 8-bit field is set to '0x81'. [2]

private_stream_id: Unused (0xFF)

PES_data_packet_header_length: This 4-bit field indicates the length in bytes of the PES_data_ private_date_bytes.

PES_data_private_data_byte: This is an 8-bit field and a more detailed usage of the area depends on a service. A receiver unit may skip this field.

Asynchronous_PES_data_byte: This is an 8-bit field contains the transmitted data.

[2] Refer to Informative Explanation 1 (1).

Chapter 6 Data carousel transmission protocol

6.1 Transmission with DSM-CC data carousel

The data carousel transmission protocol defined in this chapter is designed to implement general synchronized or asynchronous data transmission without a need of streaming data such as data download to a receiver unit or content transmission for multimedia services. The data carousel transmission protocol defined in this standard is based on the DSM-CC data carousel specified in ISO/IEC 13818-6 [3]. Transmitting data repeatedly, as defined in the DSM-CC data carousel , allows a receiver unit to obtain data on demand at anytime during a transmission period. Data is transmitted in a module unit that consists of blocks. All blocks other than that at the end of the module have the same size and each block is transmitted in sections.

According to the data carousel transmission protocol, data is transmitted using the DownloadDataBlock message (hereafter referred to as DDB message) and the DownloadInfoIndication message (hereafter referred to as DII message). The two messages are components of the User-to-Network download protocol specified in ISO/IEC 13818-6. The data body is transmitted by the DDB message with each module divided into blocks.

6.2 DownloadInfoIndication (DII) message

A DII message is part of a DSM-CC control message. Therefore, a DII message transmits message content by containing itself in the userNetworkMessage() in the DSM-CC section.

The DII message version is indicated by transaction_number in the transaction_id field of dsmccMessageHeader. This version number is common to all DII messages of the data carousel and the version number is incremented by one when content of one or more DII messages have been changed.

6.2.1 Syntax and semantics of DII message

The data structure of a DII message is shown in Table 6-1.

Table 6-1 Data structure of DownloadInfoIndication message

Syntax	Bits	Mnemonic
DownloadInfoIndication() {		
dsmccMessageHeader()		
downloadId	32	uimsbf
blockSize	16	uimsbf
windowSize	8	uimsbf
ackPeriod	8	uimsbf
tCDownloadWindow	32	uimsbf
tCDownloadScenario	32	uimsbf
compatibilityDescriptor()		
numberOfModules	16	uimsbf
for(i=0;i< numberOfModules;i++) {		
moduleId	16	uimsbf
moduleSize	32	uimsbf
moduleVersion	8	uimsbf
moduleInfoLength	8	uimsbf
for(i=0;i< moduleInfoLength;i++) {		

[3] Amendment 1 to ISO/IEC 13818-6 is at the step of FDAM as of September 1999.

moduleInfoByte	8	uimsbf
}		
}		
privateDataLength	16	uimsbf
for(i=0;i< privateDataLength;i++) {		
privateDataByte	8	uimsbf
}		
}		

Semantics of DII fields:

dsmccMessageHeader () (DSM-CC message header): As specified in Clause 6.2.2.

downloadId (Download identifier): This 32-bit field serves as a label to uniquely identify the data carousel. In the case of a data event operation, data_event_id is inserted into bits 28-31 of downloadId as shown in Figure 6-1. Otherwise, the range and values to ensure the uniqueness is specified in an operational guideline.

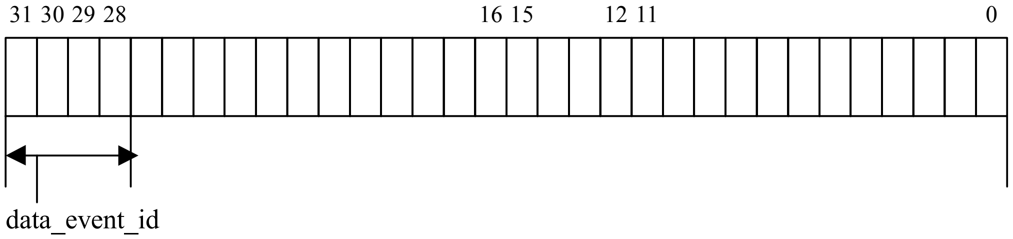


Figure 6-1 Coding of data_event_id in downloadId

data_event_id (Data event identifier): This 4-bit field of bits 28-31 in downloadId is an identifier to identify the data event, despite of being preceded and/or followed by other data event for the same service, and to allow the intended local content transmitted in the data carousel of the data event and the event message (refer to Chapter 7) to be successfully received. [4]

blockSize (Block length): This 16-bit field indicates the byte length of each block of the DDB message other than the last block of the module.

windowSize: This 8-bit field is not used for data carousel transmission and the value is set to 0.

ackPeriod: This 8-bit field is not used for data carousel transmission and the value is set to 0.

tCDownloadWindow: This 32-bit field is not used for data carousel transmission and the value is set to 0.

tCDownloadScenario: This 32-bit field indicates the timeout period in which the download is assumed to be completed in microseconds.

compatibilityDescriptor(): The compatibilityDescriptor() structure specified in ISO/IEC 13818-6 is contained in this field. When the content of the compatibilityDescriptor() structure is not needed, descriptorCount is set to 0x0000 and then the field length is 4-byte.

numberOfModules (Number of module): This 16-bit field indicates the number of the modules described in the following loop in this DII message.

moduleId (Module identifier): This 16-bit field indicates the identification of the module described in the following moduleSize, moduleVersion, and moduleInfoByte fields.

moduleSize (Module length): This 32-bit field indicates the byte length of the module. When the byte length of the module is not known, it is set to 0.

moduleVersion: This 8-bit field indicates the version of this module.

[4] At the emission side, different data_event_id is allocated to adjacent local contents.

moduleInfoLength (Module Information Length): This 8-bit field indicates the byte length of the following module information area.

moduleInfoByte (Module Information): This 8-bit unit field may be used to insert descriptors related to the module. The tag values of the descriptors to be inserted are defined in Table 6-2. These descriptors are defined in Section 6.2.3.

privateDataLength: This 16-bit field indicates the byte length of the following PrivateDataByte field.

privateDataByte (Private Data): This 8-bit unit field may be used to contain a data structure in a descriptor format. The data structure is defined based on a data coding format or by a broadcaster. The semantics of the tag values of the descriptors for this field is defined in Table 6-2. The possible descriptors for this field are those defined in Clause 6.2.3 and by a data coding format.

Table 6-2 Semantics of descriptor tags of module information area and private area in DII

Value of descriptor tag	Semantics
0x01 - 0x7F	Reserved tag values of DVB-compatible descriptors to be inserted into the module information area and the private area. (Clause 6.2.3)
0x80 - 0xBF	Available tag values of descriptors defined by a broadcaster.
0xC0 - 0xEF	Reserved for tag values of descriptors to be inserted into the module information area and the private area. (Clause 6.2.3)
0xF0 - 0xFF	Reserved tag values of descriptors defined based on a data coding format.

6.2.2 Syntax and semantics of dsmccMessageHeader()

The data structure of dsmccMessageHeader() is defined in Table 6-3.

Table 6-3 Data structure of dsmccMessageHeader

Syntax	Bits	Mnemonic
dsmccMessageHeader() {		
protocolDiscriminator	8	uimsbf
dsmccType	8	uimsbf
messageId	16	uimsbf
transaction_id	32	uimsbf
reserved	8	bslbf
adaptationLength	8	uimsbf
messageLength	16	uimsbf
if(adaptationLength>0) {		
dsmccAdaptationHeader()		
}		
}		

Semantics of **dsmccMessageHeader()** fields:

protocolDiscriminator: This 8-bit field is set to 0x11 and indicates that this message is a MPEG-2 DSM-CC message.

dsmccType (DSM-CC type): This 8-bit field indicates the type of the MPEG-2 DSM-CC message. In a DII message for data carousel transmission, it is set to 0x03 (U-N download message).

messageId (Message type identifier): This 16-bit field identifies the DSM-CC message type. In a DII message, it is set to 0x1002.

transaction_id (Transaction identifier): This 32-bit field identifies the message and has a versioning function.

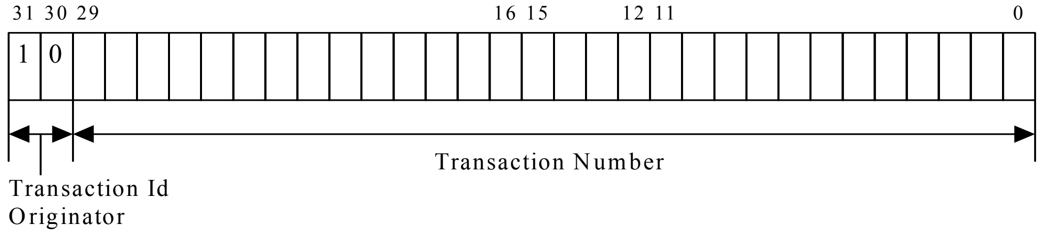


Figure 6-2 Format of transaction_id

The format of transaction_id is shown in Figure 6-2. The Transaction Number field in bits 0-29 is used to identify the version of the DII, as specified in ISO/IEC 13818-6. The value of bits 30-31 is set to '10' (transaction_id allocated by the Network) as defined in Transaction Id Originator, as specified in ISO/IEC 13818-6.

adaptationLength: This 8-bit field indicates the byte number of the dsmccAdaptationHeader() field.

messageLength: This 16-bit field indicates the number of bytes of the message immediately following this field. That is, the value is a sum of the payload length and the dsmccAdaptationHeader() length.

dsmccAdaptationHeader(): The data structure of this field is defined in Section 6.4.

6.2.3 Descriptors of module information area and private area

In this section, the descriptors used for a module information area and a private area of a DII message are defined.

The types of the descriptors used in a module information area and a private area are shown in Table 6-4. Any of these descriptors may be used in the module information area and/or the private area as needed. The descriptors contained in the private area in a DII apply to all the modules in the DII. When the module information area and the private area have the same set of descriptors, only the descriptors in the module information area are enabled to the module.

Table 6-4 Functions and tag values of descriptors used for module information area and private area

Tag value	Descriptor	Function	Module Information area	private area
0x01	Type descriptor	Module type (MIME form etc.)	O	-
0x02	Name descriptor	Module name (File name)	O	-
0x03	Info descriptor	Module information (Character type)	O	O
0x04	Module_link descriptor	Link information (module Id)	O	-
0x05	CRC32 descriptor	CRC32 of total module	O	-
0x06	Reserved for future use			
0x07	Estimateddownload time descriptor	Estimated download time (sec.)	O	O
0x08 -0x70	Reserved for future use			
0x71	Caching priority descriptor	Caching priority for a module. The data structure and semantics of this descriptor are defined in Appendix B, Part 2, ARIB STD-B23.	O	-
0x72 - 0x7F	Reserved for future use			

Tag value	Descriptor	Function	Module Information area	private area
0x80 - 0xBF	Available to a broadcaster. Any value in this range may be defined as a tag value of a descriptor.			
0xC0	Expire descriptor	The expiration limit of module	O	O
0xC1	ActivationTime descriptor	Time when module becomes active	O	O
0xC2	CompressionType descriptor	Compression algorithm when the module is compressed.	O	-
0xC3	Control descriptor	Information required to control and interpret the module.	O	-
0xC4	ProviderPrivate descriptor	Used to add proprietary information by the network provider/broadcaster.	O	O
0xC5	StoreRoot descriptor	The location of the directory that stores content for stored data service on the storage device.	O	O
0xC6	SubDirectory descriptor	The location of the subdirectory in the directory specified in StoreRoot. A subdirectory may used to store (part of) content for stored data service.	O	O
0xC7	Title descriptor	Used to indicate the title of either content or the title of the each module in the data carousel shown to end users.	O	O
0xC8	DataEncoding descriptor	Used to identify the data coding method for data of proprietary method in the carousel.	O	-
0xC9	Time-stamped TS descriptor	Defined in Clause 8.1.4.3 in Volume 2. Used to add information when MPEG-based video/audio data is transmitted in the data carousel in the time-stamped TS format.	O	-
0xCA	Root certificate descriptor.	Information required for identifying root certificates to be used for bidirectional transmission.	O	-
0xCB	Encrypt descriptor	Defined in Part 2, ARIB STD-B25. Information required for identifying and interpreting an encrypted module.	O	O
0xCC	ACG descriptor	Defined in Part 2, ARIB STD-B25. Information relating to payment groups for conditional access for playback.	O	O
0xCD - 0xED	Reserved for future use		-	-
0xEE	Reserved for the MetadataFragment descriptor defined in ARIB STD-B38. Note that this descriptor is not contained in DII.		-	-
0xEF	Reserved for the TransportLocation descriptor defined in Appendix E, Volume 2. Note that this descriptor is not contained in DII.		-	-
0xF0 - 0xFF	Reserved for tag values of descriptors defined by each coding method and used in the private area (Table 6-2)		-	O

6.2.3.1 Type descriptor

The Type descriptor (refer to Table 6-5) indicates the type of the file transmitted as a module, according to the data carousel transmission protocol based on this standard in which a single file is transmitted as a single module.

Table 6-5 Type descriptor

Syntax	Bits	Mnemonic
Type_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for(i=0; i<N; i++) {		
text_char	8	uimsbf
}		
}		

Semantics of Type_descriptor() fields:

text_char: This is an 8-bit field. The sequence in this field indicates the media type, complying with RFC1521 and RFC1590. How to specify a media type is defined for each application as follows:

- For XML-based multimedia coding method, defined in Appendix C, Part 2 of this standard-
  For conditional access method, defined in Part 2, ARIB STD-B25
- For broadcasting system based on home servers, defined in ARIB STD-B38

6.2.3.2 Name descriptor

The Name descriptor (refer to Table 6-6) indicates the name of the file transmitted as a single module, according to the data carousel transmission protocol based on this standard in which a single file is transmitted as a single module. However, when the ModuleLink descriptor exists, the Name descriptor is present only in a module of the position = 0x00 in the DII.

Table 6-6 Name descriptor

Syntax	Bits	Mnemonic
Name_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for(i=0; i<N; i++) {		
text_char	8	uimsbf
}		
}		

Semantics of Name_descriptor() fields:

text_char: This is an 8-bit field. The sequence in this field indicates the name of the file transmitted as a single module using the character code or specified in a data coding standard or an operational guideline.

6.2.3.3 Info descriptor

The Info descriptor (refer to Table 6-7) describes information related to the module.

Table 6-7 Info descriptor

Syntax	Bits	Mnemonic
info_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
ISO_639_language_code	24	bslbf
for(i=0; i<N; i++){		
text_char	8	uimsbf
}		
}		

Semantics of info_descriptor() fields:

ISO_639_language_code: This 24-bit field identifies the language used in the following text_char field. The language code is represented in three alphabetic characters as specified in ISO 639-2. Each character is coded into an 8-bit representation according to ISO 8859-1 and inserted into the 24-bit field in that order.

text_char: This is an 8-bit field. The sequence in this field indicates textual information related to the file transmitted as a single module using the character code a specified in a data coding standard or an operational guideline.

6.2.3.4 Module_link descriptor

The Module_link descriptor (refer to Table 6-8) generates a list of modules by linking with other modules. Because the length of the block number field of a DDB message is restricted to 16 bits, the maximum size of one module in data carousel transmission is 256 Mbytes. When a file larger than 256 Mbytes is transmitted, the file is divided into two or more modules before being sent, associated with this descriptor.

Table 6-8 Module_Link descriptor

Syntax	Bits	Mnemonic
module_link_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
position	8	uimsbf
moduleId	16	uimsbf
}		

Semantics of module_link_descriptor() fields:

position: This 8-bit field indicates the position relation to the linked module. “0x00” indicates that the module is located at the head of the link, “0x01” indicates the middle and “0x02” indicates the end.

moduleId: This 16-bit field is the module identification of the linked module. When the position is “0x02”, the value of this field is ignored.

6.2.3.5 CRC descriptor

The CRC descriptor (refer to Table 6-9) describes the CRC value of the whole module.

Table 6-9 CRC descriptor

Syntax	Bits	Mnemonic
CRC32_descriptor() {		
descriptor_tag	8	uimsbf

descriptor_length	8	uimsbf
CRC_32	32	rpchof
}		

Semantics of CRC32_descriptor() fields:

CRC_32: This 32-bit field stores the CRC value calculated against the whole module. The CRC value shall be calculated as defined in Appendix B, Part 2 of ARIB STD B-10.

6.2.3.6 Estimated download time descriptor

The estimated download time descriptor (refer to Table 6-10) describes an estimated period required to download the module.

Table 6-10 Estimated download time descriptor

Syntax	Bits	Mnemonic
est_download_time_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
est_download_time	32	uimsbf
}		

Semantics of est_download_time_descriptor() fields:

est_download_time: This 32-bit field indicates the estimated period (in seconds) required to download the module.

6.2.3.7 Expire descriptor

The Expire descriptor (refer to Table 6-11) describes the expiration limit of the module. For example, when the module is stored in a receiver unit having a storage device, stored data expires when the expiration limit comes. When an Expire descriptor is not available to the module, the module has no expiration limit.

Table 6-11 Expire descriptor

Syntax	Bits	Mnemonic
Expire_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
time_mode	8	uimsbf
if (time_mode == 0x01) {		
MJD_JST_time	40	bslbf
}		
else if (time_mode == 0x04) {		
reserved_future_use	8	bslbf
passed_seconds	32	uimsbf
}		
}		

Semantics of Expire_descriptor() fields:

time_mode (Time mode): This field indicates the time designating mode of the expiration limit.

time_mode	Time designation method	Semantics
0x00	-	Reserved for future use
0x01	MJD_JST_time	Absolute time indicated with modified Julian date and Japan standard time

0x02	-	Reserved for future use
0x03	-	Reserved for future use
0x04	passed_seconds	Passed time in seconds after download
0x05-0xFF	-	Reserved for future use

MJD_JST_time: This 40-bit field is coded in the case of time_mode = 0x01 and the expiration limit is indicated using JST and MJD (refer to Appendix C, Part 2 of ARIB STD B-10). This field codes the least significant 16 bits of MJD in a 16-bit representation and codes the following 24-bit area into six 4-bit binary coded decimal (BCD) representations.

passed_seconds: This 32-bit field is coded when time_mode = 0x04 and indicates the expiration limit using passed time (in seconds) after the download.

6.2.3.8 Activation Time descriptor

The Activation Time descriptor (refer to Table 6-12) indicates the time when the module content becomes valid. When this descriptor does not exist, the module becomes valid immediately after the reception.

Table 6-12 Activation Time descriptor

Syntax	Bits	Mnemonic
Activation_Time_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
time_mode	8	uimsbf
if (time_mode==0x01 || time_mode == 0x05) {		
MJD_JST_time	40	bslbf
} else if (time_mode==0x02) {		
reserved_future_use	7	bslbf
NPT_time	33	uimsbf
} else if (time_mode==0x03) {		
reserved_future_use	4	bslbf
eventRelativeTime	36	bslbf
}		
}		

Semantics of Activation_Time_descriptor() fields:

time_mode (Time mode): This 8-bit field indicates the time designation method for the expiration limit

time_mode	Time designation method	Semantics
0x00	-	Reserved for future use.
0x01	MJD_JST_time	Absolute time indicated with MJD and JST time. However, a time at playback is referred to when the stream recorded content is played back.
0x02	NPT_time	NPT (refer to Clause 7.1.1)
0x03	eventRelativeTime	Relative time designation after the program start time (in milliseconds)
0x04	-	Reserved for future use.
0x05	MJD_JST_time	Absolute time indicated with MJD and JST time. However, a time at broadcast is referred to when the stream recorded content is played back.
0x06-0xFF	-	Reserved for future use.

MJD_JST_time: This 40 bit field is coded when time_mode 0x01 or 0x05 and indicates the time with JST and MJD when module becomes valid (refer to Appendix C, Part 2 of ARIB STD B-

10). This field codes the lowest 16 bits of MJD into a 16-bit representation and codes the following 24-bit area into a six 4-bit binary coded decimal (BCD) representations.

NPT time: This 33-bit field is coded when time_mode = 0x02 and indicates the time using Normal Play Time of DSM-CC when the module becomes valid (refer to Clause 7.1.1).

event Relative Time: This 36-bit field is coded in the case of time_mode = 0x03 and indicates the time in a relative time from the program start time. The relative time is coded in the order of hour (2 digits), minute (2 digits), second (2 digits), millisecond (3 digits) using nine 4-bit binary coded decimal (BCD) representations.

6.2.3.9 Compression Type descriptor

The Compression Type descriptor (refer to Table 6-13) indicates that the module has been compressed and indicates its compression algorithm and the module size in bytes before compression. A module that has not been compressed does not have this descriptor.

Table 6-13 Compression Type descriptor

Syntax	Bits	Mnemonic
Compression_Type_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
compression_type	8	uimsbf
original_size	32	uimsbf
}		

Semantics of Compression_Type_descriptor() fields:

Compression_type: This 8-bit field designates the compression type used to compress the module. The value to identify the compression type is specified in an operational guideline.

original_size: This 32-bit field indicates the module size in bytes before compression.

6.2.3.10 Control descriptor

The Control descriptor (refer to Table 6-14) describes information necessary to interpret and control the module.

Table 6-14 Control descriptor

Syntax	Bits	Mnemonic
Control_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0; i++; i<N) {		
control_data_byte	8	bslbf
}		

Semantics of Control_descriptor() fields:

Control_data_byte: This is an 8-bit field. This field indicates information necessary to interpret and control the module by inserting data structures defined in the data coding method or other related rules such as an operational guideline.

6.2.3.11 Provider Private descriptor

The Provider Private descriptor (refer to Table 6-15) allows a network provider/broadcaster to add proprietary information according to the rules for each scope.

Table 6-15 Provider Private descriptor

Syntax	Bits	Mnemonic
Provider_Private_descriptor () {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
private_scope_type	8	bslbf
scope_identifier	32	bslbf
for (i = 0; i< N; i++) {		
private_byte	8	bslbf
}		
}		

Semantics of Provider_Private_descriptor() fields:

private_scope_type: This is an 8-bit field. This field indicates the type of the identifier representing the scope of this descriptor.

scope_identifier: This is a 32-bit field. This field indicates the scope identifiers for each scope type.

private_scope_type	scope_identifier	Bits	Mnemonic	Semantics
0x00	-	-	-	Reserved for future use
0x01	network_id	16	uimsbf	Network identifier is used as the scope for this descriptor.
	padding	16	bslbf	
0x02	network_id	16	uimsbf	Service identifier is used as the scope for this descriptor.
	service_id	16	uimsbf	
0x03	network_id	16	uimsbf	Broadcaster identifier is used as the scope for this descriptor.
	broadcaster_id	8	uimsbf	
	padding	8	bslbf	
0x04	passed_seconds	16	uimsbf	Bouquet identifier is used as the scope for this descriptor.
	padding	16	bslbf	
0x05	information_provider_id	16	uimsbf	Information provider identifier is used as s the scope for this descriptor.
	padding	16	bslbf	
0x06	CA_system_id	16	uimsbf	Conditional access system identifier is used as the scope for this descriptor.
	padding	16	bslbf	
0x07-0xFF	-	-	-	Reserved for future use

Note padding: all bits must be set to 1.

private_byte: This is an 8-bit field. This field provides additional information based on the rule specified for the scope.

6.2.3.12 StoreRoot descriptor

The StoreRoot descriptor (refer to Table 6-16) is used in a data carousel that transmits content for stored data services. This descriptor indicates the highest level directory of the storage device, in which the modules in the carousel are to be stored. This descriptor also indicates whether the specified part of the existing content is deleted or not before the modules in the data carousel are stored. Only one StoreRoot descriptor is provided in a whole data carousel and is contained in the private area.

Table 6-16 StoreRoot descriptor

Syntax	Bits	Mnemonic
store_root_descriptor () {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf

Syntax	Bits	Mnemonic
update_type	1	bslbf
reserved	7	bslbf
for (i = 0;i< N;i++) {		
store_root_path	8	uimsbf
}		
}		

Semantics of store_root_descriptor() fields:

update_type: This is a 1-bit field. This field indicates whether or not the content in the directory specified in the store_root_path field is deleted before the modules in the data carousel are stored. When update_type is 1, the directory content is deleted before it is stored. When update_type is 0, the directory content is not deleted and the modules are added to the existing content in the directory.

store_root_path: This is an 8-bit field. This field indicates the highest level directory of directories in the storage device, which stores the modules of the data carousel, using a character coding that is defined in Chapter 9 in Volume 2.

6.2.3.13 Subdirectory descriptor

The Subdirectory descriptor (refer to Table 6-17) is used in a data carousel that transmits content for stored data services. This descriptor indicates the subdirectories that store the modules of the data carousel, located in the directory specified in the StoreRoot in the storage device. When the intended subdirectory has been specified for all the modules, this descriptor is present in the private area. When separate subdirectories are allocated to separate modules, this descriptor is present in the module information area. For the module of which subdirectory is instructed by the descriptors in both area, the descriptor in the module information area is valid.

Table 6-17 Subdirectory descriptor

Syntax	Bits	Mnemonic
subdirectory_descriptor () {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i = 0;i< N;i++) {		
subdirectory_path	8	uimsbf
}		
}		

Semantics of subdirectory_descriptor() fields:

subdirectory_path: This is an 8-bit field. This field indicates the subdirectories that store the modules of the data carousel, located in the directory specified in the StoreRoot in the storage device, using a character coding that is defined in Chapter 9 in Volume 2.

6.2.3.14 Title descriptor

When the Title descriptor (refer to Table 6-18) is present in the private area, the character strings in this descriptor serve as the title of the entire content transmitted in the data carousel. When the Title descriptor is present in the module information area, the character strings in this descriptor serve as separate titles of separate modules. In either case, the titles are available to end users.

Table 6-18 Title descriptor

Syntax	Bits	Mnemonic
title_descriptor(){		

Syntax	Bits	Mnemonic
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
ISO_639_language_code	24	bslbf
for(i=0; i<N; i++){		
text_char	8	uimsbf
}		
}		

Semantics of title_descriptor() fields:

ISO_639_language_code: This 24-bit field identifies the language used in the following text_char field. The language code is specified using the corresponding three alphabet characters as specified in ISO639-2. Each alphabetical character is an 8-bit representation based on ISO8859-1 and placed in this 24-bit field in that order.

text_char: This is an 8-bit field. The sequence of this field provides the title of the entire content or the titles of the modules, that are available to end users, described using character codes specified in the data coding method or in an operational guideline.

6.2.3.15 DataEncoding descriptor

When two or more data coding methods are employed in the modules of a data carousel, the DataEncoding descriptor is used to identify each method.

Table 6-19 DataEncoding descriptor

Syntax	Bits	Mnemonic
data_encoding_descriptor() {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
data_compoent_id	16	uimsbf
for (i=0; i<N; i++) {		
additional_data_encoding_info	8	uimsbf
}		
}		

Semantics of data_encoding_descriptor() fields:

data_component_id: This 16-bit field is used for used for indicating individual data coding method. The field must contain the same value as the data_component_id in the data_component_descriptor(), that is specified in the Notification No.37 of Ministry of Internal Affairs and Communications in 2003.

additional_data_encoding_info: This 8-bit field is used to extend identifications or to contain additional information for each data coding method.

6.2.3.16 Root certificate descriptor

The root certificate descriptor (See Table 6-20) contains information required for identifying root certificates used for bidirectional transmission. Operation of a root certificate such as identification and storage are defined in Appendix 1, Volume 2.

Table 6-20 Root certificate descriptor

Syntax	Bits	Mnemonic
root_cetificate_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf

root_certificate_type	1	bslbf
reserved	7	bslbf
if (root_certificate_type == 0){		
for (i=0; i<N; i++){		
root_certificate_id	32	uimsbf
root_certificate _version	32	uimsbf
}		
} else {		
for (i=0; i<N; i++){		
reserved	64	bslbf
}		
}		
}		

Semantics of root_certificate_descriptor:

root_certificate_type: This 1-bit field is used for identifying a type of a root certificate. When this field is '1', the root certificate is identified as a broadcaster-specific certificate. When this field is '0', the root certificate is identified as a generic certificate. The terms 'broadcaster-specific certificate' and 'generic certificate' are defined in Appendix 1, Volume 2.

root_certificate_id: This 32-bit field indicates whether or not the root certificate required to be stored in a root certificate storage area whose details are defined in an operational guideline exists or not. This field also contains information for identifying the root certificate. When this field is '0xFFFFFFFF ', it indicates that no root certificate to be stored in an appropriate root certificate storage area exists in the module. When this field contains the value other than '0xFFFFFFFF ', it indicates that the root certificate to be stored in an appropriate root certificate storage area exists in the module. The value used for this purpose is defined in an operational guideline.

root_certificate_version: This 32-bit field indicates the version of the root certificate identified with root_certificate_id. The values available to this field are defined in an operational guideline. Note that when the root_certificate_id field is '0xFFFFFFFF', the root_certificate_version field is set to '0xFFFFFFFF'.

6.3 DownloadDataBlock (DDB) message

The content of a DDB message is transmitted by storing in the downloadDataMessage() field in the DSM-CC section.

6.3.1 Syntax and semantics of DDB message

A DDB message is the data structure for transmitting data blocks (Table 6-21). A module is divided into data blocks with a fixed length. In this case, each block is represented with a block number in the DDB message to allow a receiver unit to reassemble the blocks in the intended order.

As is specified in ISO/IEC 13818-6, when DDB messages are transmitted in MPEG-2 TS, only the DDB messages having the same downloadId must be included in the same PID packets. This means that DDB messages in two different carousels must not present in a single elementary stream.

Table 6-21 Data structure of Download Data Block

Syntax	Bits	Mnemonic
DownloadDataBlock() {		
dsmccDownloadDataHeader()		
moduleId	16	uimsbf
moduleVersion	8	uimsbf
reserved	8	bslbf
blockNumber	16	uimsbf
for(i=0;i<N;i++) {		

Syntax	Bits	Mnemonic

blockDataByte
    }
}
	8	uimsbf

Semantics of DDB() fields:

moduleId: This is a 16-bit field and indicates the identification number of the module, to which this block belongs.

moduleVersion: This is an 8-bit field and indicates the version of the module, to which this block belongs.

blockNumber: This is a 16-bit field and indicates the position of this block within the module. The first block of a module must be represented by the block number 0.

blockDataByte: This is an 8-bit field. The size of a series of the block data area is the same as the block size described in the DII, that is, the size of the blocks divided from a module. However, the block of the last blockNumber in the module may be smaller than the block size described in DII.

6.3.2 Syntax and semantics of dsmccDownloadDataHeader()

The data structure of the dsmccDownloadDataHeader() is defined in Table 6-22.

Table 6-22 Data structure of dsmccDownloadDataHeader

Syntax	Bits	Mnemonic
dsmccDownloadDataHeader() {		
protocolDiscriminator	8	uimsbf
dsmccType	8	uimsbf
messageId	16	uimsbf
downloadId	32	uimsbf
reserved	8	bslbf
adaptationLength	8	uimsbf
messageLength	16	uimsbf
if(adaptationLength>0) {		
dsmccAdaptationHeader()		
}		
}		

Semantics of dsmcc DownloadDataHeader() fields:

protocol Discriminator: This 8-bit field is set to 0x11 to indicate that this message is a MPEG-2 DSM-CC message.

dsmccType: This 8-bit field indicates the type of the MPEG-2 DSM-CC message and is set to 0x03 (U-N download message) for the DDB message in data carousel transmission.

messageId: This 16-bit field identifies the type of the DSM-CC message and is set to 0x1003 for a DDB message.

downloadId: This 32-bit field is set to the same value as the download identifier in the corresponding DII message.

adaptaionLength: This 8-bit field indicates the number of bytes of the dsmccAdaptationHeader() field.

messageLength: This 16-bit field indicates the length of the message in bytes from the field immediately following this field. The value is identical to the sum of the payload length and the dsmccAdaptationHeader length.

dsmccAdaptationHeader(): The data structure of this field is defined in Section 6.4.

6.4 Syntax of dsmccAdaptationHeader()

In dsmccMessageHeader() which is the header part of a DII message and dsmccDownloadDataHeader() which is the header part of a DDB message, the common data structure dsmccAdaptationHeader() can be placed.

The data structure of dsmccAdaptationHeader is shown in Table 6-23.

Table 6-23 Data structure of dsmccAdaptationHeader()

Syntax	Bits	Mnemonic
dsmccAdaptationHeader() {		
adaptationType	8	uimsbf
for(i = 0; i < (adaptationLength - 1); i++){		
adaptationDataByte	8	uimsbf
}		
}		
}		

Semantics of dsmccAdaptationHeader() fields:

adaptationType: This 8-bit field indicates the type of the adaptation. This field value indicates an adaptation format as shown in the following table.

AdaptationType	Adaptation format	Definition in ISO/IEC 13818-6 (reference)
0x00	Reserved	Same as in the left column
0x01	Reserved	DSM-CC Conditional Access
0x02	Reserved	DSM-CC user identifier
0x03	Reserved	Same as in the left column
0x04-0x7F	Reserved	Same as in the left column
0x80-0xFF	User defined	Same as in the left column

The following applies to the adaptation types used in this standard:

- Operation of user defined adaptation format of adaptation type 0x80 - 0xFF is optionally operated by a broadcaster.

6.5 Syntax of DSM-CC section

DII messages and DDB messages are transmitted using DSM-CC sections shown in Table 6-24.

Table 6-24 DSM-CC section (Transmission of DII/DDB messages)

Syntax	Bits	Mnemonic
DSMCC_section() {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
private_indicator	1	bslbf
reserved	2	bslbf
dsmcc_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
if (table_id == 0x3B) {		
userNetworkMessage()		
}		

```
    else if (table_id == 0x3C) {
        downloadDataMessage()
    }
    else if (table_id == 0x3E) {
        for (i=0;i<dsmcc_section_length-9;i++) {
            private_data_byte                          8       uimsbf
        }
    }
    if(section_syntax_indicator == ‘0’) {
        Checksum                                       32      uimsbf
    }
    else {
        CRC_32                                         32      rpchof
    }
}
```

Semantics of DSMCC_section() fields:

table_id: This 8-bit field contains the number identifying the type of data in the DSM-CC section payload. Based on the value in this field, a coding rule specific to the following field in the DSM-CC section is applied. The table identification values are shown in the following table, as specified in ISO/IEC 13818-6.

table_id	DSM-CC section type	(reference) Definition of ISO/IEC 13818-6
0x3A	Reserved	Multi-protocol encapsulation
0x3B	DII message (this Chapter)	U-N message including DII
0x3C	DDB message (this Chapter)	Same as in the left column
0x3D	Stream descriptor (Chapter 7)	Same as in the left column
0x3E	Private data	Same as in the left column
0x3F	Reserved	Same as in the left column

section_syntax_indicator: This is a 1-bit field. When it is set to 1, it indicates that a CRC32 exists at the end of the section. When it is set to 0, it indicates that check sum exists. It must be set to 1 to transmit DII messages and DDB messages.

private_indicator: This 1-bit field stores the complement value of the section_syntax_indicator.

dsmcc_section_length: This 12-bit field indicates the number of bytes of the area from the beginning of the field immediately following this field to the end of the section. The value in this field does not exceed 4093.

table_id_extension: This 16-bit field is set as shown below according to the table_id:

- When the value of the table_id equals 0x3B, this field conveys the least significant two bytes of the transaction_id.
- When the value of the table_id equals 0x3C, this field conveys the module_id.

version_number: This 5-bit field is set as shown below according to the table_id:

- When the value of tabel_id equals 0x3B, this field is set to 0. [5]
- When the value of table_id equals 0x3C, this field is set to the least significant 5 bits of the moduleVersion.

current_next_indicator: This is a 1-bitflag. When it is set to "1", it indicates that the sub-table is the current sub-table. When it is set to "0", the sub-table being sent is not applied yet and used as the next sub-table. When the value of table_id is in the range from 0x3A to 0x3C, this field is set to "1".

section_number: This 8-bit field indicates the section number. When the section contains a DII message, this field is set to be 0. When this section contains a DDB message, this field conveys the least significant eight bits of the block number of the DDB.

[5] Version management of DII message is made by transaction_id.

last_section_number: This 8-bit field indicates the last section number (the section which has the maximum section number) of the sub-table to which the section belongs.

userNetworkMessage(): The DII message is stored

downloadDataMessage(): The DDB message is stored.

Chapter 7 Event message transmission protocol

The event message transmission protocol provides a means to send message information immediately or at a specified time for the application running on a receiver unit from a broadcast station.

The event message transmission protocol defined in this chapter extends the specification of stream descriptor and its DSM-CC section transmission protocol specified in ISO/IEC 13818-6 to deal with various time appointing methods required for the applications.

7.1 Stream descriptor

The stream descriptors used in this standard are shown in Table 7-1.

Table 7-1 Stream descriptor and tag value

Tag value	Descriptor	Function
0x00 - 0x16	Reserved for future use	
0x17	NPTReferenceDescriptor	To describe relation of NPT and STC
0x18 - 0x3F	Reserved for future use	
0x40	General_event_descriptor	To describe data contents and synchronizing information of event message
0x41 - 0x7F	Reserved for future use	
0x80 - 0xFF	Selectable range for tag value of broadcaster defined descriptor	

7.1.1 NPT reference descriptor

The NPT reference descriptor (NPTReferenceDescriptor) is a descriptor to communicate the relation between NPT (Normal Play Time) and STC (System Time Clock).

Table 7-2 NPT reference descriptor

Syntax	Bits	Mnemonic
NPTReferenceDescriptor () {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
postDiscontinuityIndicator	1	bsblf
dsm_contentId	7	uimsbf
reserved	7	bsblf
STC_Reference	33	uimsbf
reserved	31	bsblf
NPT_Reference	33	tcimsbf
scaleNumerator	16	tcimsbf
scaleDenominator	16	tcimsbf
}		

Semantics of NPTReferenceDescriptor() fields:

postDiscontinuityIndicator: When this 1-bit field is 0, it indicates that this descriptor is valid immediately after the reception. When it is 1, it indicates that this descriptor is valid in the next 'system time base discontinuity.'

dsm_contentId: This 7-bit field stores an identifier to specify the NPT reference identifier related to which NPT time base.

STC_Reference: This 33-bit field stores a System Time Clock (STC) value corresponding to the NPT_Reference value. It is given by the following formula:

$$\text{STC\_Reference}^{k} = (\text{STC}_{\text{NPT(k)}} / 300) \ \% \ 2^{33}$$

Where, STC$_{\mathrm{NPT}\,(k)}$ is the value of System Time Clock (STC) when NPT is the NPT_Reference value.

NPT_Reference: This 33-bit field indicates a NPT value when System Time Clock (STC) is the STC_Reference value.

scaleNumerator: This 16-bit field stores a value given by the following formula, indicating the rate of NPT advance:

scaleNumerator = NPT_Reference$_j$-NPT_Reference$_i$

scaleDenominator: This 16-bit field stores a value given by the following formula indicating rate of STC advance:

scaleDenominator = STC_Reference$_j$-STC_Reference$_i$

The values and semantics of scaleNumerator and scaleDenominatorare are shown in the following table.

scaleNumerator	scaleDenominator	Semantic
0	0	Indicates that scaleNumerator and scaleDenominator are not used.
0	Other than 0	NPT continues constant value irrelevant to STC
1	1	NPT and STC advance at the same rate.
Other than 0	0	(Such combination shall not be used.)

7.1.2 General event descriptor

The general event descriptor (General_event_descriptor) is a descriptor to communicate information applied generally to event messages.

The data structure of the general event descriptor is shown in Table 7-3.

Table 7-3 General event descriptor

Syntax	Bits	Mnemonic
General_event_descriptor () {		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
event_msg_group_id	12	uimsbf
reserved_future_use	4	bslbf
time_mode	8	uimsbf
if (time_mode == 0) {		
reserved_future_use	40	bslbf
} else if (time_mode == 0x01 || time_mode == 0x05) {		
event_msg_MJD_JST_time	40	bslbf
} else if (time_mode == 0x02) {		
reserved_future_use	7	bslbf
event_msg_NPT	33	uimsbf
} else if (time_mode == 0x03) {		
reserved_future_use	4	bslbf
event_msg_relativeTime	36	bslbf
}		
event_msg_type	8	uimsbf
event_msg_id	16	uimsbf
for (i=0;i<N;i++) {		
private_data_byte	8	uimsbf
}		
}		

Semantics of General_event_descriptor() fields:

event_msg_group_id (Message group identifier): This 12-bit field indicates the identifier to identify the message group to be received by the application. Details of operations are specified for each data component identifier. When operating an event message having more than one message group identifier at the same time, only general event descriptors having the same message group identifier is included in one DSM-CC section.

time_mode (Time mode): This 8-bit field indicates the method to designate the time when the event message is occurred. Refer to the following table.

time_mode	Time designation method	Semantic
0x00	None	Event message is occurred immediately after reception.
0x01	MJD_JST_time	Event message is occurred at the absolute time indicated by MJD and JST time. Event message is also occurred when playing back stream recorded contents by referring to the time of playing back.
0x02	NPT	Event message is occurred at the time specified with the NPT time data.
0x03	eventRelativeTime	Event message is occurred when the period specified in this field (in milliseconds) after the program start time.
0x04	-	Reserved for future use.
0x05	MJD_JST_time	Event message is occurred at the absolute time indicated by MJD and JST time. When playing back stream recorded contents, event message is occurred by referring to the on air time.
0x06-0xFF	-	Reserved for future use.

event_msg_MJD_JST_time: This 40-bit field is coded in the case of time_mode = 0x01 and indicates the time when the event message is generated in Japan Standard Time (JST) and Modified Julian Date (MJD) (refer to Appendix C, Part 2 of ARIB STD B-10). This field conveys the least significant 16 bits of MJD followed by six 4-bit binary coded decimal (BCD) representations.

event_msg_NPT: This 33-bit field is coded in the case of time_mode = 0x02 and indicates the time when the event message is occurred, using Normal Play Time of DSM-CC (refer to Clause 7.1.1).

event_msg_relativeTime: This 36-bit field is coded in the case of time_mode = 0x03 and indicates that the event message is occurred when the period specified in this field passes after the program start time. The value of this field is described in the order of hour (2 digits), minute (2 digits), second (2 digits), and millisecond (3 digits) to form nine 4-bit binary coded decimal (BCD) representations.

event_msg_type (Message type): An identifier indicating the event message type. Usage and semantics is specified in each data coding method.

event_msg_id (Message identifier): This 16-bit field contains the identifier to identify each event message. Usage and semantics is specified in each data coding method.

private_data_byte (Private data): This is an 8-bit field. This field stores the information related to the event message defined as to event_msg_type specified for each data coding method.

7.2 Syntax of DSM-CC section transmitting stream descriptor

The stream descriptor is transmitted in the DSM-CC section shown in Table 7-4.

Table 7-4 DSM-CC section (Transmission of stream descriptor)

Syntax	Bits	Mnemonic
DSMCC_section() {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
private_indicator	1	bslbf
reserved	2	bslbf
dsmcc_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
if (table_id == 0x3D) {		
for (i=0; i<N; i++) {		
stream_descriptor ()		
}		
}		
if(section_syntax_indicator == ‘0’) {		
Checksum	32	uimsbf
}		
else {		
CRC_32	32	rpchof
}		
}		

Semantics of DSMCC_section() fields:

table_id (Table identifier): This 8-bit field is set to 0x3D to indicate that the stream descriptor is stored in the payload of the DSM-CC section.

section_syntax_indicator: This is a 1-bit field. When it is 1, it indicates that CRC32 exists at the end of the section. When it is 0, it indicates that check sum exists. To transmit an event message, this field must be set to 1.

private_indicator: This 1-bit field stores the complement value of the section_syntax_indicator.

dsmcc_section_length: This 12-bit field indicates the byte length of the area from the position immediately following this field to the end of the section. This field value does not exceed 4093.

data_event_id: This 4-bit field is the identifier to identify the successive data events that use the event message and to allow for the intended local content delivered by the data events to receive the correct event message. Successively transmitted local contents are allocated with different identifiers.

event_msg_group_id (Message group identifier): This 12-bit field contains the identifier to identify the group of event messages to be received by the application. Detailed semantics is specified in each data coding method.

version_number: This 5-bit field is the version number of the sub-table. The version number is incremented by 1 when any piece of information in the sub-table has been changed. The available values are from 0 to 31.The value 0 is used to update the value 31.

current_next_indicator: This is a 1-bit flag. When it is set to '1', it indicates that the sub-table is the current sub-table. When it is set to '0', the sub-table being sent is not yet applied and used as the next sub-table.

section_number: This 8-bit field indicates the section number.

last_section_number: This 8-bit field indicates the last section number (that is the section having the maximum section number) of the sub-table to which the concerned section belongs.

Chapter 8 Interaction channel protocols for digital broadcasting

8.1 Data transfer protocols

A protocol used over public networks including PSTNs, ISDNs, and mobile networks for bidirectional interactive services, consist of the following three phases:

- The Line connection and disconnection phase;
- The Data Link establishment and termination phase;
- The Data transmission phase.

Note: The Data Link establishment and termination phase is only used for the direct connection model with shared access point. (Informative Explanation 5.(1) 2))

8.1.1 Line connection and disconnection phase

In this phase, a receiver connects to and disconnects to a center over a public network. The line connection and disconnection are controlled with AT commands from a modem, a Terminal Adapter, or a data communications adapter for mobile phone/PHS communications services.

8.1.2 DataLink establishment and termination phase

In the Data Link establishment phase, which starts immediately after the line connection has been completed, a data transmission link is established between the receiver and a host at the center. In the Data Link termination phase, after the data transmission has been completed, the link between the receiver and the host is terminated. The Data link establishment and termination phase can be applied to data transmission protocols, which do not specify destination address for each data packet in the data link layer. Therefore, this phase is applicable to all protocols in sections 8.1.3 and 8.1.5.

When basic modems are used, the layer (e.g. the physical layer (MNP4), the data link layer, the network layer) at which error detection and correction protocols is used is specified in an operational guideline.

	Protocol stack	
Application layer	Selected according to service	
Data link layer	Protocol conforming to part of X.28 (it requires function for specifying host No.)	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC*2: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis *3
PHS*1	PIAFS:32kbit/s or more.	Same as left column

*1: Personal Handy-phone System: ARIB RCR STD-28 “PERSONAL HANDY PHONE SYSTEM RCR STANDARD”

*2: Personal Digital Cellular: ARIB RCR STD-27 “PERSONAL DIGITAL CELLULAR TELECOMMUNICATION SYSTEM RCR STANDARD”

*3: Converted to analogue data in mobile phone network (same hereinafter).

8.1.3 Protocols used to direct connections (data transmission phase)

8.1.3.1 Basic functions (protocols)

In the following protocols, the layer at which error detection and correction protocols are performed, that is, the physical layer (MNP4), the data link layer, or the network layer, is specified in an operational guideline.

Note: The phrase "protocols at the Physical layer" in this document means protocols of the Physical layer and Transport layer in the ITU-T Rec. J.111 and J.113. The phrase "protocols at the Data link layer and higher layers" means the network independent protocols in the Rec. J.111. These differences comes from the IP protocol's layer. In this specification, the IP is stated as a protocol at the network layer, while the IP is assumed to be at the higher medium layer in the J.111.

- Text Communications Protocol Stack

	Protocol stack	
Application layer	Selected according to services	
Data link layer	Non Procedure(TTY protocol)	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis
PHS	PIAFS:32kbit/s or more.	Same as left column

- Communications Protocol Stacks for Binary Transmissions

	Protocol stack	
Application layer	Selected according to services	
Data link layer	BASIC Procedures (only required functions implemented) Code-independent mode	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis
PHS	PIAFS:32kbit/s or more.	Same as left column

	Protocol stack	
Application layer	Selected according to services	
Data link layer	BASIC Procedure s (JIS X5002) Code-independent mode	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis
PHS	PIAFS:32kbit/s or more.	Same as left column

	Protocol stack	
Application layer	Selected according to services	
Data link layer	PPP in HDLC-like Framing (RFC1662)	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis
PHS	PIAFS:32kbit/s or more.	Same as left column

	Protocol stack	
Application layer	Selected according to services	
Data link layer	HDLC protocol (ISO 3309,ISO 4335,ISO 7809)	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis
PHS	PIAFS: 32kbit/s or more.	Same as left column

	Protocol stack	
Application layer Transport layer	HTTP1.0(RFC1945)subset is used.	
Network layer	-	
Data link layer	-	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600bit/s	PDC: 9600bit/s or V.32bis + V.42bis
PHS	PIAFS:32kbit/s or more.	Same as left column

- Protocols for Internet and Intranet Connections

	Protocol stack	
Application layer	Selected according to services from HTTP1.0 (RFC1945), FTP, POP3, SMTP, etc.	
Transport layer	TCP(RFC793), UDP(RFC768)	
Network layer	IP(RFC791)/ICMP(RFC792)	
	Receiver	Host
Data link layer	PPP(RFC1661, 1662)/IPCP(RFC1332)	PPP(RFC1661, 1662)/IPCP(RFC1332) [*1]
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column
Mobile phone (circuit switched service)	PDC: 9600 bit/s	PDC: 9600 bit/s or V.32 bis+ V.42 bis

	Protocol stack	
Mobile phone (packet switched service)	PDC-P, etc.: 9600bit/s or more.	Packet communications network *2 (leased line/ISDN, etc.) 9600bit/s or more.
PHS	PIAFS:32kbit/s or more.	Same as left column

*1: Not implemented at the host in the case of mobile phones (packet service)

*2: Physical layer protocol and data transmission rate are converted in mobile phone network.

8.1.3.2 Advanced functions (protocols)

When transmitting content in addition to the basic functions, it is preferable to use the following protocols, which are used over the Internet.

- Using PSTN

	Protocol stack	
Application layer	Selected according to services from HTTP1.0 (RFC1945), HTTP1.1(RFC2616),Telnet, FTP, NNTP, SMTP, POP3, DNS (RFC1123), etc.	
Transport layer	TCP(RFC793), UDP(RFC768)	
Network layer	IP(RFC791)/ICMP(RFC792)	
Data link layer	PPP(RFC1661, 1662)/IPCP(RFC1332) PAP(RFC1334)/CHAP(RFC1994), PPP Internet Protocol Control Protocol Extensions for Name Server Addresses(RFC1877), CCP (RFC1962)	
Physical layer	Receiver	Host
Basic modem	V.22bis or later	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column

- Using Mobile Phones and PHS

	Protocol stack	
Application layer	Selected according to services from HTTP1.0 (RFC1945), HTTP1.1(RFC2616),Telnet, FTP, NNTP, SMTP, POP3, DNS (RFC1123), etc.	
Transport layer	TCP(RFC793), UDP(RFC768)	
Network layer	IP(RFC791)/ICMP(RFC792)	
	Receiver	Host
Data link layer	PPP(RFC1661, 1662) /IPCP(RFC1332) PAP(RFC1334)/CHAP(RFC1994), PPP Internet Protocol Control Protocol Extensions for Name Server Addresses(RFC1877) LCP Extension(RFC1570), CCP (RFC1962)	PPP(RFC1661, 1662) /IPCP(RFC1332)*1 PAP(RFC1334)/CHAP(RFC1994), PPP Internet Protocol Control Protocol Extensions for Name Server Addresses(RFC1877) LCP Extension(RFC1570), CCP (RFC1962)
Physical layer		
Mobile phone (circuit switched service)	RCR STD-27(PDC) ARIB STD-T53(CDMA Cellular System)	PDC CDMA Cellular System

	Protocol stack	
Mobile phone (packet service)	RCR STD-27(PDC-P)	Packet communications network *2 (leased line/ISDN, etc.)
Mobile phone (packet service)	ARIB STD-T53(CDMA Cellular System)	Packet communications network *2 (leased line/ISDN, etc.)
Mobile phone (packet/circuit switched service)	ARIB STD-T63(IMT2000 DS-CDMA)	(leased line/ISDN, etc.)
Mobile phone (packet service)	ARIB STD-T64(IMT2000 MC-CDMA)	Packet communications network *2 (leased line/ISDN, etc.)
PHS	PIAFS	Same as left column

*1: Not implemented at the host in the case of mobile phones (packet service)

*2: Physical layer protocol and data transmission rate are converted in mobile phone network.

- Using ISDN

Channel classification	B channel	D channel	
Layer	Protocol stack	Protocol stack	
Application layer	Selected according to service from HTTP1.0 (RFC1945), HTTP1.1(RFC2616),Telnet, FTP, NNTP, SMTP, POP3, DNS (RFC1123), etc.	Selected according to services	
Transport layer	TCP(RFC793) , UDP(RFC768)	-	
Network layer	IP(RFC791)/ICMP(RFC792)	TTC JT-Q.931	X.25 (packet level) *1
Data link layer	PPP(RFC1661,1662)/IPCP(RFC1332) PAP(RFC1334)/CHAP(RFC1994), PPP Internet Protocol Control Protocol Extensions for Name Server Addresses(RFC1877) , CCP (RFC1962)	TTC JT-Q.921	
Physical layer	TTC JT-I.430	TTC JT-I.430	

*1: Used in the Dch "packet call control phase"

- Using Ethernet (use of ISDN, ADSL, FTTH and others as return channel)

Layer	Protocol stack
Application layer	Selected according to services from HTTP1.0 (RFC1945), HTTP1.1(RFC2616),Telnet, FTP, NNTP, SMTP, POP3, DNS (RFC1123), etc.
Transport layer	TCP(RFC793) , UDP(RFC768)
Network layer	IP(RFC791)/ICMP(RFC792)
Data link layer	PPP(RFC1661,1662)/PPPoE(RFC2516) /IPCP(RFC1332) , CCP (RFC1962) *1 PAP(RFC1334)/CHAP(RFC1994), *1 PPP Internet Protocol Control Protocol Extensions for Name Server Addresses(RFC1877) *1 IEEE802.2/ARP(RFC826)
Physical layer	IEEE802.3,IEEE802.11

*1: Choice of actual protocol depends on supported PPP-related protocol in a DSU, an ADSL modem, a router, or other working device. It also depends on how network services are applied.

8.1.4 Protocols for massive calls reception service

AT commands which are built into a modem or TA, shall be available to the receiver. The application shall provide functions to control the AT commands.

Layer	ISDN protocol stack
Application layer	Selected according to services
Physical layer	TTC JT-I.430

	PSTN protocol stack	
Application layer	Selected according to services	
Physical layer	Receiver	Host
Basic modem	V.22bis or later.	Same as left column
Advanced modem	V.34 or later + V.42bis	Same as left column

8.1.5 Protocols for interaction channel carrying request and broadcast channel carrying response

Layer	Protocol stack (Broadcast channel)	Protocol stack (Interaction channel)
Application layer	Selected according to services	
Transport layer	TCP(RFC793), UDP(RFC768)	
Network layer	IP(RFC791)/ICMP(RFC792)	
Data link layer	DSM-CC Section for Private Data (EN301192), etc.	Various protocols, as appropriate for PSTN basic functions, PSTN advanced functions, mobile telephone packet services, ISDN advanced functions, PHS and mobile telephones
Physical layer	MPEG-2 TS	

8.2 Security

Various security functions are needed to support a wide range of services. A receiver shall equip with one of the two security functions sets.

Basic functions set: This set requires to having cipher functions including secret key cryptography. A receiver with the basic function set must obtain, in secret, a key that is shared with an intended party for encrypted communication before the intended transmission begins.

Advanced functions set: In addition to basic function set, this set requires to having public key cryptography, message digests, and certificates (that are issued by a Certification Authority to authorize a public key for an implementation of public key cryptography). The existence of a public key implies that a receiver with the advanced functions set is ready to share a secret key for secret key cryptography. There is no need for a receiver with the advanced functions set to obtain a secret key in advance. More specifically, TLS 1.0 (see RFC 2246 [C17]) is recommended to be used, which is also available to communication via the Internet.

The sections following section 8.2.1 in this chapter are two-part sections to describe each set. Note that for encryption and identification to be used in the basic functions set, encryption functions and identification card functions implemented for the conditional access system are allowed to be used as required.

8.2.1 Data encryption

To encrypt a set of information, a specific encryption algorithm of secret key cryptography and public key cryptography, which is selected from among various algorithms, is to be used. Although it is not recommended that multiple algorithms are used for a single service, each receiver is recommended to have expandability to support future revisions of this standard. To be expandable enough, a receiver should support a selection of encryption algorithms. To implement this selection, there are several options including methods for informing which encryption algorithm is used (for more information, see "Specification of abstract syntax notation one [ASN.1]" ITU-T X.208 [C1], ITU-T X.209 [C2]) and methods for showing equipped encryption algorithms and negotiating to decide an algorithm to be used (methods similar to those defined in ITU-T H.234 [C3]).

When a system employing secret key cryptography and/or public key cryptography is designed and operated, how involved keys are managed must be considered carefully. Guidelines on key management must be developed depending on the required security level (For more information, see Informative Explanation).

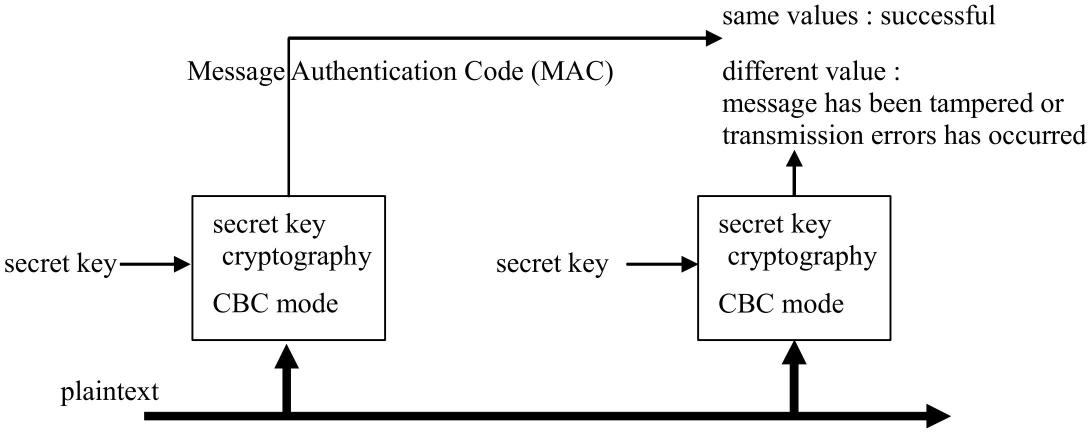
Informative Explanation 6 (1) describes some samples of public key cryptographies registered in JIS X 5060 [C4], which is characterized by encryption algorithms and strength of encryption. Informative Explanation 6 (3) introduces a sample of a simpler cryptography implementation that is suitable for occasions where strength of cryptography is less important. Selecting this type of implementation implies considerations for risks of being decrypted.

8.2.2 Data integrity

- Basic functions set: Secret key cryptography is employed.

  A Message Authentication Code (MAC) is allowed to replace a secret key cryptography implementation. For more information, see JIS X 5055 [C5].

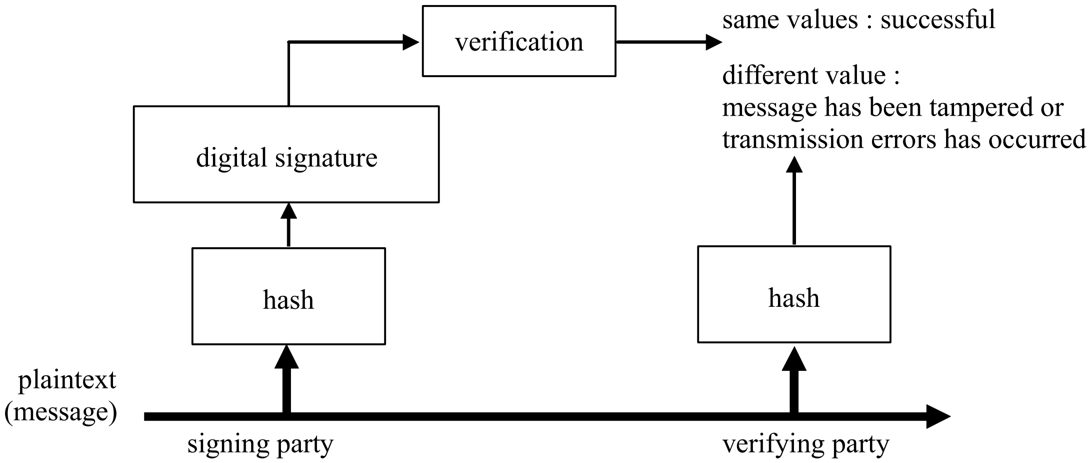
  These methods are employed in order to ensure that an intended message arrives at an intended recipient without being tampered nor transmission errors. That is, these methods are not operated for a purpose of encrypted transmission itself. These methods are responsible for transmitting a message body and encrypting a whole message in CBC mode. When the encrypting process finishes, a IV register value is transmitted as a MAC. The recipient performs the equivalent operation. If the message has been tampered or any transmission error has occurred, the two MAC values differ, allowing an abnormality to be detected.



As a simpler method, a Cyclic Redundancy Check (CRC) algorithm is allowed to be used. Note that CRC does not provide a way of detecting that an incoming message has been tampered.

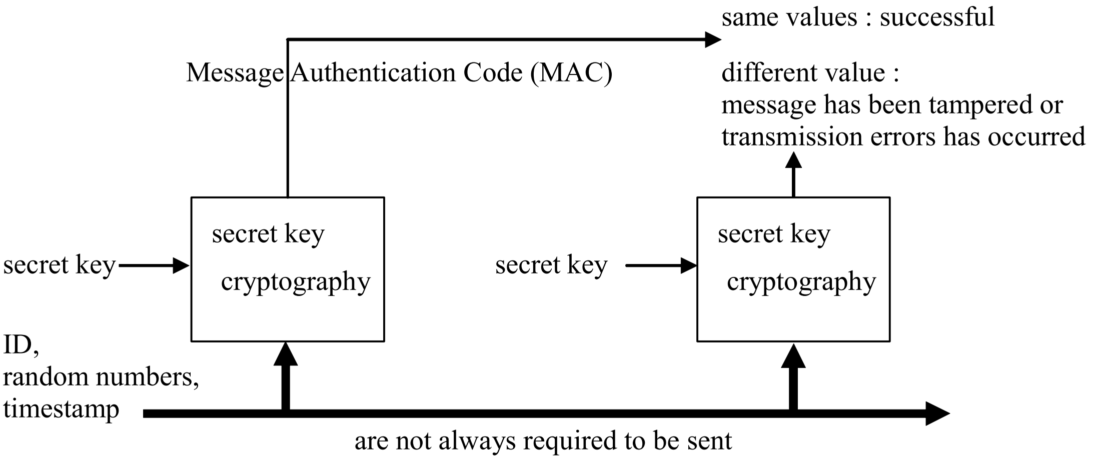
- Advanced functions set: Public key cryptography and a message digest algorithm are employed.

A message digest is created from a message to be sent and then a digital signature is also generated. A message digest is also known as a hash function (see JIS X 5057 [C6, C7]) and is designed to generate a specific length of a summary (digest) from an arbitrary length of a message. Any digital signature has a maximum length. To generate a digital signature for a rather long message more efficiently, a message digest is generated from a message to be sent as a preprocessing step, and a digital signature, in turn, is generated from the message digest (see JIS X 5056-3 [C8]). A verifying party compares a result of applying the message digest to the received message with a result of verifying the message with the digital signature. When the verifying party finds the two values equal, the data integrity is assured.



8.2.3 Authentication of communicating party

- Basic functions set: Secret key cryptography is employed (message recovery method).



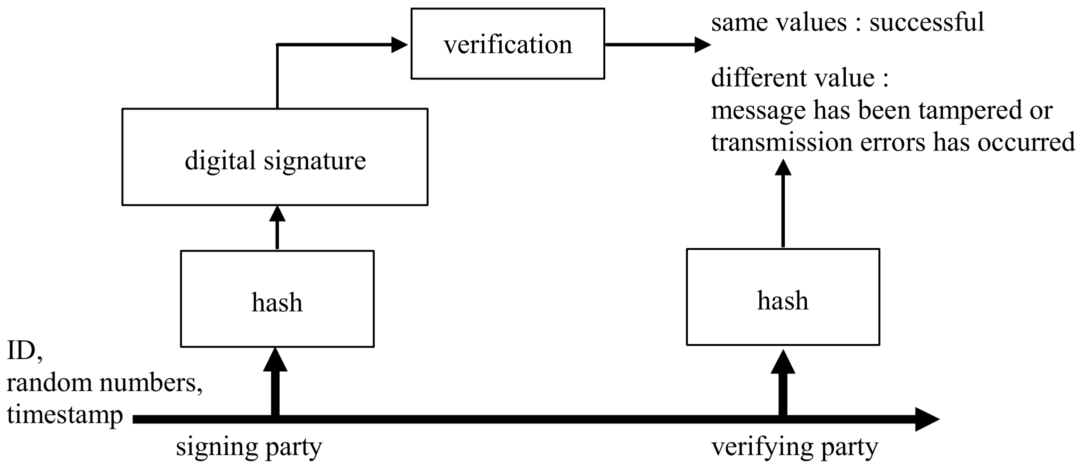
When a sending party and a verifying party shares a secret key prior to sending a message, the secret key can be used by the sending party to encrypt a message. The authenticity of the sending party can be assured as long as the verifying party decodes the encrypted message to find that the decrypted plaintext is meaningful.

There is a way of authenticating each other. First, a verifying party adds one (1) to a random number that has been generated by a sending party or performs other simple operations that are agreed on by the two parties in advance. Second, the verifying party encrypts the decoded message again and sends it back to the sending party. Finally, the sending party can assure the authenticity of the verifying party by performing the same operation as the verifying party did.

(Depending on a required security level, a sender ID function of a massive call reception service or other functions may be used to authenticate a communicating party as a simpler way.)

- Advanced functions set: Public key cryptography is employed.

To authenticate a communicating party, a certificate issued by a Certification Authority under a public key cryptosystem (see X.509 [C9]) is obtained and is verified by using a public key.



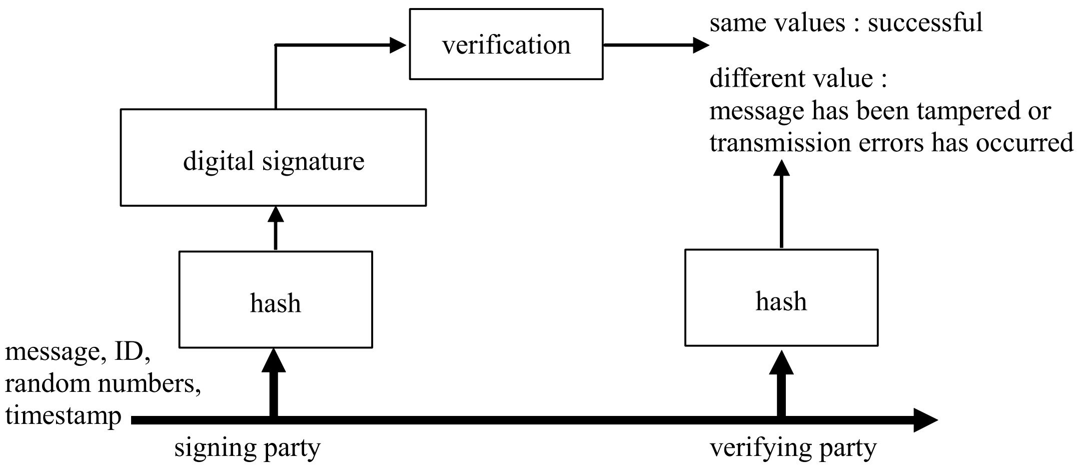
(To perform authentication in a simpler way, a hash function may be used as a one-way function for a simple authentication implementation, as defined in X.509.)

8.2.4 Signature

- Basic functions set: Secret key cryptography is employed.

A Message Authentication Code, as described in section 8.2.2, is applied to a message that otherwise would require a digital signature.

- Advanced functions set: Public key cryptography and a message digest algorithm are employed.



A message digest is generated from a message to be sent, and then a digital signature under a public key cryptosystem is applied.

Chapter 9 Operational standard for identifiers

Allocation of each identifier used in service information shall be as shown in table 9-1. Range of value in the table includes reserved value, which will be specified in the future.

Therefore, operator/provider defined specification may contain reserved values; however, it shall be registered and released as the operator/provider defined signal including the reserved values.

Table 9-1 Operational standard for identifiers

Identifiers	Corresponding portion of Vol. 3, STD-B24		Bit	Range of value	Type of definition	Remarks
	Section	Contained in				
Stream type (stream_type)	4	PMT	8	0x00 – 0x7F	Specified by Ministry of Internal Affairs and Communications	Specified by the Notification
				0xC0 – 0xFF	Specified by standardization organization	Registered and released after deliberation
				0x80 – 0xBF	Specified and operated by the company	
Stream identifier (stream_id)	5.1 and others	PES packet	8	0xBC – 0xFF	Specified by Ministry of Internal Affairs and Communications	Specified by the Notification
data_identifier	5.1 and others	Synchronized PES, Asynchronous PES	8	0x00 – 0x7F	–	Defined in DVB, ATSC, and DAVIC
				0x80 – 0xFF	Specified by standardization organization	Defined as the user-defined area in DVB, ATSC, and DAVIC
private_stream_id	5.1 and others	Synchronized PES, Asynchronous PES	8		Specified by standardization organization	Registered and released after deliberation
Download identifier (downloadId)	6.2.1 and others	DII, DDB	32	Bits 28 – 31 under a data event operation	Specified by standardization organization	Allocated
				Others	Specified and operated by the company	
Data event identifier (data_event_id)	6.2.1 and others	DII, DSM-CC section that transmits stream descriptors	4		Specified and operated by the company	
Module identifier (moduleId)	6.2.1 and others	DII, DDB, Module_Link descriptor	16		Specified and operated by the company	
Module version (moduleVersion)	6.2.1 and others	DII, DDB	8		Specified and operated by the company	
protocolDiscriminator	6.2.2 and others	DII, DDB	8		Specified by Ministry of Internal Affairs and Communications	Specified by the Notification
DSM-CC type (dsmccType)	6.2.2 and others	DII, DDB	8		Specified by Ministry of Internal Affairs and Communications	Specified by the Notification
Message type identifier (messageId)	6.2.2 and others	DII, DDB	16		Specified by Ministry of Internal Affairs and Communications	Specified by the Notification

transaction identifier (transaction_id)	6.2.2 and others	DII	32	Bits 30 – 31	Specified by standardization organization	Allocated
				Bits 0 – 29	Specified and operated by the company	
descriptor tag contained in DII (descriptor_tag)	6.2.3	DII	8	0x01– 0x7F	Specified by standardization organization	Registered and released after deliberation (Reserved for the compatibility with DVB)
				0x80 – 0xBF	Specified and operated by the company	
				0xC0 – 0xFF	Specified by standardization organization	Registered and released after deliberation
ISO_639_language_code	6.2.3.3 and others	Info descriptor, Title descriptor	24		Specified by standardization organization	Comply with the specification of international standardization organization
Position	6.2.3.4	Module_Link descriptor	8		Specified by standardization organization	Registered and released after deliberation (To be compatible with DVB)
Time mode (time_mode)	6.2.3.7	Expire descriptor	8		Specified by standardization organization	Registered and released after deliberation
	6.2.3.8	ActivationTime descriptor	8		Specified by standardization organization	Registered and released after deliberation
	7.1.2	General event descriptor	8		Specified by standardization organization	Registered and released after deliberation
compression_type	6.2.3.9	CompressionType descriptor	8		Specified and operated by the company	Registered and released
private_scope_type	6.2.3.11	ProviderPrivate descriptor	8		Specified by standardization organization	Registered and released after deliberation
scope_identifier	6.2.3.11	ProviderPrivate descriptor	32		Specified by standardization organization	Registered and released after deliberation (To be specified for each private_scope_type)
Data component identifier (data_component_id)	6.2.3.15	DataEncoding descriptor	16		Specified by standardization organization	Registered and released on application (Refer to Annex J in Part 2 of STD-B10)
root_certificate_id	6.2.3.16	Root certificate descriptor	32	0xFFFFFFFF	Specified by standardization organization	When a module does not store a certificate to be stored.
				Values other than 0xFFFFFFFF	Specified and operated by the company	When a module stores a certificate to be stored.
root_certificate_version	6.2.3.16	Root certificate descriptor	32	0xFFFFFFFF	Specified by standardization organization	When root_certificate_id is 0xFFFFFFFF.
				Values other than 0xFFFFFFFF	Specified and operated by the company	

adaptation type (adaptationType)	6.4	dsmccAdaptationHeader	8	0x00 – 0x7F	Specified by standardization organization	Registered and released after deliberation (Comply with the specification of international standardization organization)
				0x80 – 0xFF	Specified and operated by the company	
Table identifier *1 (table_id)	6.5	DSM-CC section	8	0x00 – 0x41, 0x82 – 0x85,0xFF	Specified by Ministry of Internal Affairs and Communications	Specified by the Notification
				0x42 – 0x81, 0x86 – 0x8F, 0xC0 – 0xFE	Specified by standardization organization	Registered and released after deliberation
				0x90 – 0xBF	Specified and operated by the company	
Table identifier extension (table_id_extension)	6.5	DSM-CC section	16		Specified by standardization organization	Registered and released after deliberation (Comply with the specification of international standardization organization)
Version number (version_number)	6.5	DSM-CC section	5		Specified by standardization organization	Registered and released after deliberation (Comply with the specification of international standardization organization)
Section number (section_number)	6.5	DSM-CC section	8		Specified by standardization organization	Comply with the specification of international standardization organization
Stream descriptor tag (descriptor_tag)	7.1	DSM-CC section	8	0x00 – 0x7F	Specified by standardization organization	Registered and released after deliberation (Comply with the specification of international standardization organization)
				0x80 – 0xFF	Specified and operated by the company	
dsm_contentId	7.1.1	NPT reference descriptor	7		Specified and operated by the company	
Message group identifier (event_msg_group_id)	7.1.2 and others	General event descriptor, DSM-CC section	12		Specified and operated by the company	Specified for each data coding method
Message type (event_msg_type)	7.1.2	General event descriptor	8		Specified and operated by the company	Specified for each data coding method
Message identifier (event_msg_id)	7.1.2	General event descriptor	16		Specified and operated by the company	Specified for each data coding method

*1 See clause 5.2 in Part 1 of ARIB STD-B10.

Annex A Video PES (normative)

When using video PES of MPEG-2 Video (ISO/IEC 13818-2) to transmit data, the user data area (User Data field) following the picture header of a video stream shall be used. The syntax of the User Data field is shown in Table A-1. A more detailed usage of this area depends on broadcasters.

Table A-1 Syntax of user data field of video stream

Syntax	Bits	Mnemonic
User Data() {		
user_data_start_code	32	bslbf
while (nextbits() != 0x000001) {		
user_data	8	uimsbf
}		
next_start_code()		
}		

user_data_start_code: 0x000001b2

Annex B Audio PES (normative)

B.1 Data transmission format of audio PES coded with MPEG-2 BC Audio

When using audio PES of MPEG-2 BC Audio (ISO/IEC 13818-3) to transmit data, the ancillary data area, which may contain data other than MPEG audio for each audio frame, shall be used. The syntax of the ancillary data area is shown in Table B-1. A more detailed usage of this area depends on broadcasters.

Table B-1 Ancillary data area of audio stream (MPEG-2 BC Audio)

Syntax	Bits	Mnemonic
MPEG1_ancillary_data() {		
if (ext_bit_stream_present == 1) {		
for (b=0; b<8*n_ad_bytes; b++)		
ancillary_bit	1	bslbf
}		
}		

B.2 Data transmission format of audio PES coded with MPEG-2 AAC Audio

When using audio PES of MPEG-2 AAC Audio (ISO/IEC 13818-7) to transmit data, the data_stream_element area, which may contain data other than audio data for each raw_data_block shall be used. The syntax of this area is shown in Table B-2. A more detailed usage of this area depends on broadcasters.

Table B-2 Data stream element area of audio stream (MPEG-2 AAC Audio)

Syntax	Bits	Mnemonic
data_stream_element() {		
element_instance_tag	4	uimsbf
data_byte_align_flag	1	uimsbf
cnt = count	8	uimsbf
if (cnt == 255) cnt += esc_count	8	uimsbf
if (data_byte_align_flag)		
byte_alignment()		
for (i=0; i<cnt; i++)		
data_stream_byte[element_instance_tag][i]	8	uimsbf
}		

Annex C PSI/SI information for data carousel transmission and event message transmission (normative)

For the data coding method applied to data transmission based on the data carousel and event message scheme, an additional syntax to be inserted in the field depending on data coding method in data_component_descriptor in PMT and data_content_descriptor in EIT specified in ARIB STD B-10 is defined in this Annex.

This definition is based on the following assumptions about transmission operation for data broadcasting services.

- The DII and DDB belonging to one carousel are transmitted in one ES.
- One data broadcasting service may consist of two or more carousels. Event messages may be transmitted.

C.1 Content of additional_data_component_info loop of data_component_descriptor

To insert the information for reception control of the data carousel and the event messages in the loop which contains additional_data_component_info at the end of data_component_descriptor, the following data structure is placed in the loop, as specified by the data coding method.

Syntax	Bits	Mnemonic
additional_arib_carousel_info(){		
data_event_id	4	uimsbf
event_section_flag	1	bslbf
reserved	3	bslbf
}		

Semantics of additional_arib_carousel_info() fields:

data_event_id: This is a 4-bit identifier. This identifier recognizes the preceding and following data events using the data carousel and the event messages to avoid failing to receive the appropriate local content transmitted in the data carousel and the event messages. In the case of all the bits set to 1, it means that the DIIs and the event messages having a data_event_id identifier of any value delivered in this service are valid.

event_section_flag: This 1-bit field indicates whether or not event messages are sent with this component.

0: Event messages are not transmitted

1: Event messages are transmitted

reserved: Reserved.

C.2 Selector byte of data_contents_descriptor

To insert the information for data carousel reception control in the selector byte of the data content descriptor in EIT and others, the following data structure is placed in the selector_byte field, as specified by the concerned data coding method..

C.2.1 Data structure for data carousel reception control for non-stored data services

For non-stored data services such as real-time reception, the following data structure is inserted.

Syntax	Bits	Mnemonic
arib_carousel_info(){		
num_of_carousels	8	uimsbf

for(i=0; i< num_of_carousels; i++) {		
component_tag	8	uimsbf
event_section_flag	1	bslbf
reserved_future_use	3	bslbf
component_size_flag	1	bslbf
default_transaction_id_flag	1	bslbf
default_timeout_DII_flag	1	bslbf
default_leak_rate_flag	1	bslbf
if (component_size_flag == ‘1’) {		
component_size	32	uimsbf
}		
if (default_transaction_id_flag == ‘1’) {		
transaction_id	32	uimsbf
}		
if (default_timeout_DII_flag == ‘1’) {		
timeout_value_DII	32	uimsbf
}		
if (default_leak_rate_flag == ‘1’) {		
leak_rate	22	uimsbf
reserved	2	bslbf
}		
}		
}		

Semantics of arib_carousel_info() fields:

num_of carousels: This 8-bit field indicates the number of carousels included in the following loop.

component_tag: This 8-bit field designates the component stream transmitting the carousels with the component tag given by the stream identifier descriptor in PMT.

event_section_flag: This field indicates whether or not event messages are sent with this component.

component_size_flag: This 1-bit field indicates whether or not the data structure contains the component size. When the component_size field value is not available, it is not coded.

0: not coded

1: coded

default_transaction_id_flag: This 1-bit field indicates whether or not the transaction identifier is coded in this syntax. To designate to gain DII of arbitrary transaction identification, transaction identifier is not coded.

0: not coded

1: coded

default_timeout DII_flag: This 1-bit field indicates whether or not the DII timeout value is coded in this syntax. When default value specified in the operation guideline is used as DII timeout value, it is not coded.

0: not coded

1: coded

default_leak_rate_flag: This 1-bit field indicates whether or not leak rate is coded in this syntax. When default value specified in the operation guideline is used as leak rate value, it is not coded.

0: not coded

1: coded

component_size: This 32-bit field indicates the total size (in bytes) of the data transmitted in the carousels in this component.

transaction_id: The field indicates the DII transaction identification value transmitted in this component. When the transaction identifier is not coded, it means that a DII with arbitrary transaction identifier is to be gained.

time_out_value_DII: This 32-bit field indicates the recommended timeout value (in milliseconds) to receive the whole section of the DII of this carousel. When the value is 0xFFFFFFFF, it means that no recommended timeout value exists.

leak_rate: This 22-bit field indicates the leak rate of the transport buffer in the receiver unit in a unit of 50 byte/s.

C.2.2 Data structure for data carousel reception control for stored data services

For stored data services, the following data structure is inserted.

Syntax	Bits	Mnemonic
arib_ stored_carousel_info(){		
num_of_carousels	8	uimsbf
for(i=0; i< num_of_carousels; i++) {		
component_tag	8	uimsbf
num_dataEvent_flag	1	bslbf
num_modules_flag	1	bslbf
num_resources_flag	1	bslbf
compressed_component_size_flag	1	bslbf
component_size_flag	1	bslbf
default_transaction_id_flag	1	bslbf
default_timeout_DII_flag	1	bslbf
default_leak_rate_flag	1	bslbf
if (num_dataEvent_flag == ‘1’) {		
num_dataEvent	16	uimsbf
}		
if (num_modules_flag == ‘1’) {		
num_modules	32	uimsbf
}		
if (num_resources_flag == ‘1’) {		
num_resources	32	uimsbf
}		
if (compressed_component_size_flag == ‘1’) {		
compressed_component_size	32	uimsbf
}		
if (component_size_flag == ‘1’) {		
component_size	32	uimsbf
}		
if (default_transaction_id_flag == ‘1’) {		
transaction_id	32	uimsbf
}		
if (default_timeout_DII_flag == ‘1’) {		
timeout_value_DII	32	uimsbf
}		
if (default_leak_rate_flag == ‘1’) {		
leak_rate	22	uimsbf
reserved	2	bslbf
}		
}		
}		

Semantics of arib_ stored_carousel_info() fields:

num_of carousels: This 8-bit field indicates the number of carousels included in the following loop.

component_tag: This 8-bit field designates the component stream transmitting the carousels with the component tag given by the stream identifier descriptor in PMT.

num_dataEvent_flag: This 1-bit field indicates whether or not the data structure contains the number of data events. When the num_dataEvent field value is not available, it is not coded.

0: not coded

1: coded

num_modules_flag:This 1-bit field indicates whether or not the data structure contains the total number of the modules. When the num_modules field value is not available, it is not coded.

0: not coded

1: coded

num_resources_flag: This 1-bit field indicates whether or not the data structure contains the total number of the resources. When the num_resources field value is not available, it is not coded.

0: not coded

1: coded

compressed_component_size_flag: This 1-bit field indicates whether or not the data structure contains the compressed component size. When the compressed_component_size field value is not available, it is not coded.

0: not coded

1: coded

component_size_flag: This 1-bit field indicates whether or not the data structure contains the component size. When the component_size field value is not available, it is not coded.

0: not coded

1: coded

default_transaction_id_flag: This 1-bit field indicates whether or not the transaction identifier is coded in this syntax. To designate to gain DII of arbitrary transaction identifier, transaction identifier is not coded.

0: not coded

1: coded

default_timeout DII_flag: This 1-bit field indicates whether or not the DII timeout value is coded in this syntax. When the default value specified in the operation is used as DII timeout value, it is not coded.

0: not coded

1: coded

default_leak_rate_flag: This 1-bit field indicates whether or not the leak rate is coded in this syntax. When the default value specified in the operation is used as leak rate value, it is not coded.

0: not coded

1: coded

num_dataEvent: This 32-bit field indicates the total number of the data events in the concerned component.

num_modules: This 32-bit field indicates the total number of the modules in all the data events in the concerned component.

num_resources: This 32-bit field indicates the total number of the resources in all the modules in all the data events in the concerned component.

compressed_component_size: This 32-bit field indicates the total size (in bytes) of the data in all the data events in the data carousels in this component. Note that the size of a compressed module is calculated based on the compressed state.

component_size: This 32-bit field indicates the total size (in bytes) of the data in all the data events in the data carousels in this component. Note that the size of a compressed module is calculated based on the uncompressed state.

transaction_id: This field indicates the DII transaction identification value transmitted in this component. When the transaction identifier is not present, it means that a DII with arbitrary transaction identifier is to be gained.

time_out_value_DII: This 32-bit field indicates the recommended timeout value (in milliseconds) to receive the whole section of the DII in this carousel. When the value is 0xFFFFFFFF, it means that t no recommended timeout value exists.

leak_rate: This 22-bit field indicates the leak rate of the transport buffer in the receiver unit in a unit of 50 byte/s.

Informative Explanation

1 Supplementary notes on PES transmission protocol

(1) Value of data_identifier in the independent PES transmissions protocol

A data_identifier is present at the start of the PES_packet_data_byte to identify the data type under each of DVB EN301 192, ATSC DVS-161, and DAVIC Part 9. However, in Japan, the Notification No.37 of the Ministry of Internal Affairs and Communications in 2003 stipulates the use of data_component_id in PMT.

As a result, in order to ensure conformity with DVB, ATSC, and DAVIC, the data_identifier field in this standard contains the value allocated to the user-defined area in those standards.

(2) Features and suitable applications of PES transmission protocols

The reasons for employing the independent PES transmission protocol as a standard PES transmission method are as follows:

- Less constraints on size and more flexible.
- Video and audio data can be separately produced before being multiplexed for easier operation.
- Data conveyed by this protocol can be shared in multiple pieces of video and audio data for easier access.

In contrast, in the case of employing video PES and audio PES, although it is easy to obtain data through separate video and audio processes, the data size constraints are more demanding and accessing shared data is more difficult.

2 Supplementary notes on data carousel transmission protocol

Data carousel transmission protocol is based on the User to Network download (DSM-CC data carousel) stipulated in ISO/IEC 13818-6, to which the following have been additionally specified:

- In relation to the module data area, assuming its use for file transmissions, etc., usage stipulations in the description format conforming to DVB prEN301 192 is added. Expire, ActivationTime, and CompressionType descriptors are also added, which are not specified in DVB prEN301 192.

In addition to the download services assumed when the original ISO/IEC 13818-6 was developed, these stipulations enable the use of data carousel transmission protocol that offers efficient transmission and minimal reception processing loads in a wide variety of applications such as multimedia services.

3 Relationship among local content, content, and data content descriptor

The relationship between local content and content is described as follows, which are defined in Chapter 3:

- A content constituting a multimedia service or caption service consists of one or more local contents.
- In a content, multiple local contents could be sent in sequence of one ES. For example, by switching data events in a content, it would be possible to automatically switch a series of multimedia contents within a program.
- In contrast to contents, which have the same duration and start time as those of the event, local contents are allowed to remain after the end of the content. As a result, it is possible to continue part of the multimedia service after the end of the program.

The relationship between these concepts and the data content descriptor is described below with reference to Figure 1.

- The data_content_descriptor provides the information related to the content, which is the aggregate of the local contents at the time of that event, instead of information specific to each local content.. This ensures that the broadcast time of the content is the same as the time defined as the event.
- A local content that spans more than one event is handled as the shared content by the temporally successive two contents. However, the content that can be viewed in the next event may not be reached from the content in the next event. In this case, the latter content does not have data_content_descriptor.
- Each local content is accessed via links provided by multimedia coding such as BML. It is essential to ensure that any link is established between local contents that are in transmission to prevent from an access to a local content that is not actually broadcast.

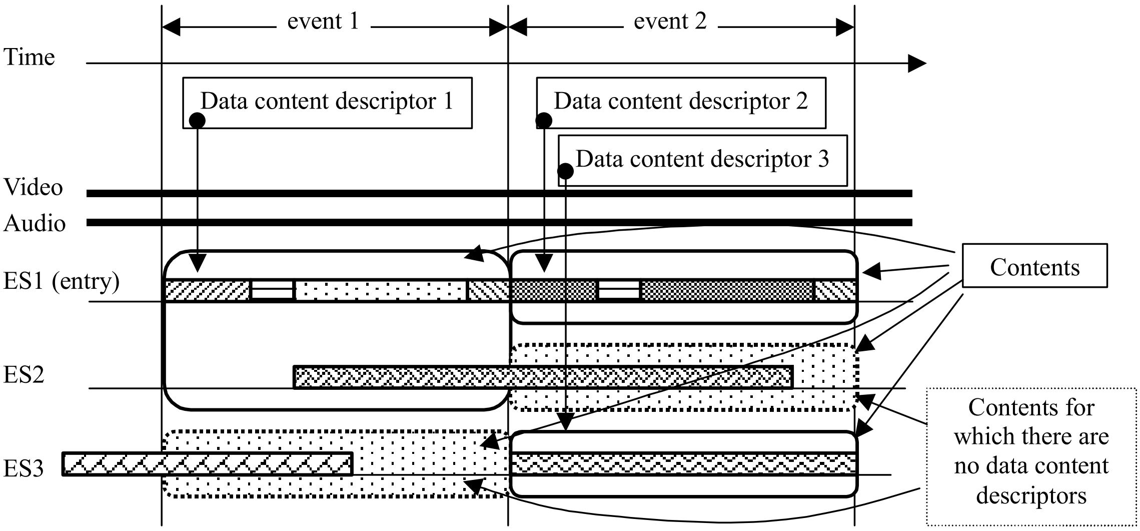

event 1
event 2
Time
Data content descriptor 1
Data content descriptor 2
Data content descriptor 3
Video
Audio
ES1 (entry)
ES2
ES3
Contents
Contents for which there are no data content descriptors

Figure 1 Relationship between local content and data content descriptors

4 Applications of StoreRoot/Subdirectory descriptors for stored data services

(1) Structure of a data carousel and structure of directories on a storage device

Each set of local contents for a stored data service is stored according to the following guidelines.

- Local contents are stored in the directory specified in StoreRoot descriptor.
- Modules in a local content are stored in the subdirectory in the above mentioned directory, as specified in SubDirectory descriptor.

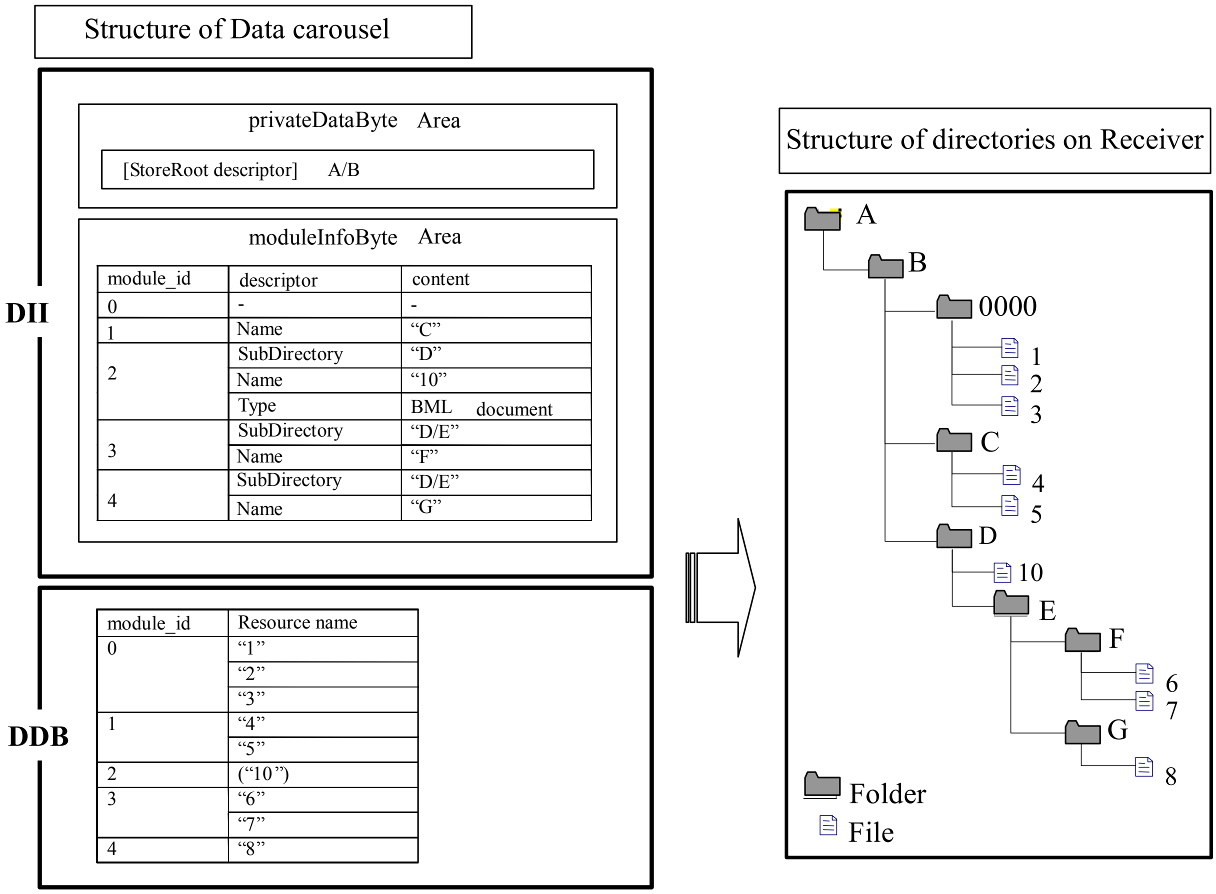
A module to be stored is mapped to the intended file on the storage device as follows:

- When a resource is directly mapped to a module:
  - The module is recognized as a file on the storage device.
  - The file name is as specified in the Name descriptor.
  - When the Name descriptor is not available, a moduleId is used as the file name.
- When resources are contained in a module in the entity format:
  - The module is recognized as a directory on the storage device and the resources in the module are recognized as the files in the directory.

- The string specified in the Name descriptor is used as the directory name.
- When the Name descriptor is not available, a moduleId is used as the directory name.
- The resource name of each resource is used as the directory name.

When the Name descriptor is not available and the moduleId is used as a directory name or a file name, the name is represented by a string with the hexadecimal notation. Note that prefix or suffix to identify the hexadecimal notation such as “0x” and “h” are not required. Instead, 0s preceding the string, as required, are given to allow the name to be a fixed length, a 4-character string. For example, when a moduleId is 0x0001, the folder name or the file name is 0001.

The relation among the StoreRoot descriptor, the Subdirectory descriptor, and the resource names is shown in the following figure.


Structure of Data carousel
DII
privateDataByte Area
[StoreRoot descriptor] A/B
moduleInfoByte Area
DDB
Structure of directories on Receiver
A
B
0000
1
2
3
C
4
5
D
10
E
F
6
7
G
8
Folder
File

module_id	descriptor	content
0	-	-
1	Name	“C”
2	SubDirectory	“D”
	Name	“10”
	Type	BML document
3	SubDirectory	“D/E”
	Name	“F”
4	SubDirectory	“D/E”
	Name	“G”

module_id	Resource name
0	“1”
	“2”
	“3”
1	“4”
	“5”
2	(“10”)
3	“6”
	“7”
4	“8”

5 Supplementary notes on interactive transmission systems

(1) Connection models

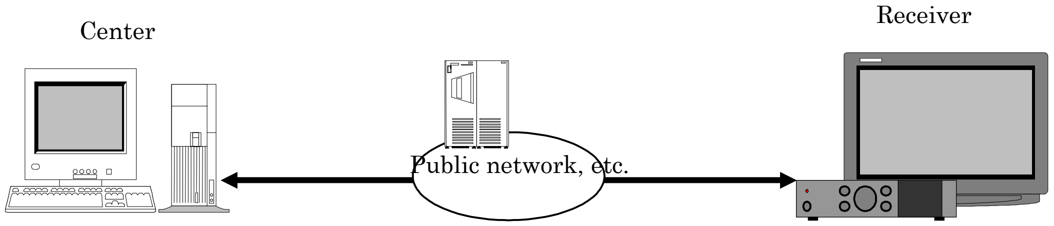
Five typical connection models for an interactive transmission system are described below.

- Symmetric bidirectional connection models

1) Direct connection model: A center and a receiver are directly connected over a public network or others.

Advantages: A receiver can have simple implementation by selecting an appropriate set of protocols.

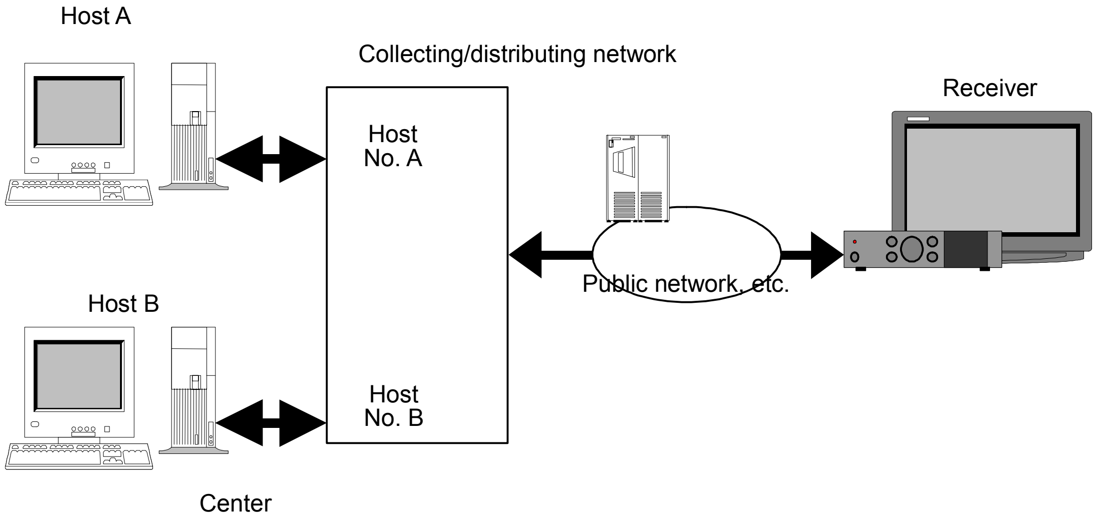
Disadvantages: A center must ensure that access points specific to the center are available.



2) Direct connection model: Through a public network or others, a receiver and a center are directly connected depending on a receiver and an application.

Advantages: A receiver can have simple implementation by selecting an appropriate set of protocols.
Centers can share access points.

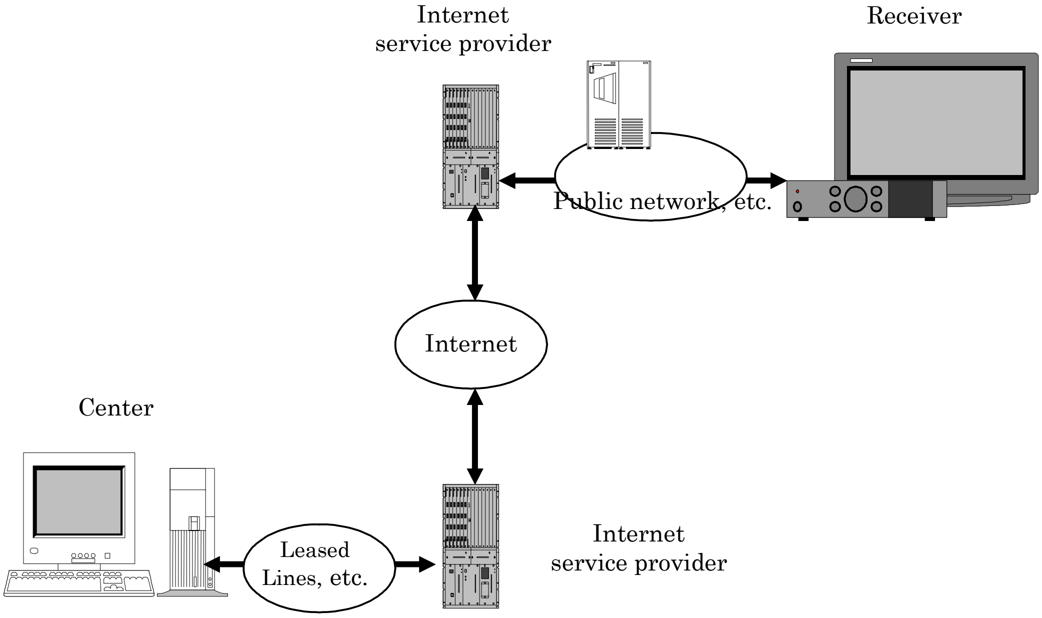
Disadvantages: Because access points are shared by more than one center, scheduling of access points may be required.



3) Internet connection: A receiver is connected via a public network or others to an access point of an Internet Service Provider (ISP). Then, the receiver is connected via the Internet to another ISP, to which a center is connected over a leased line or others.

Advantages: Existing access points across the nation can be used.

Disadvantages: A receiver must support the TCP/IP, PPP, and ISP connection protocols. In order to use services provided by a center, an end user must subscribe to an ISP.

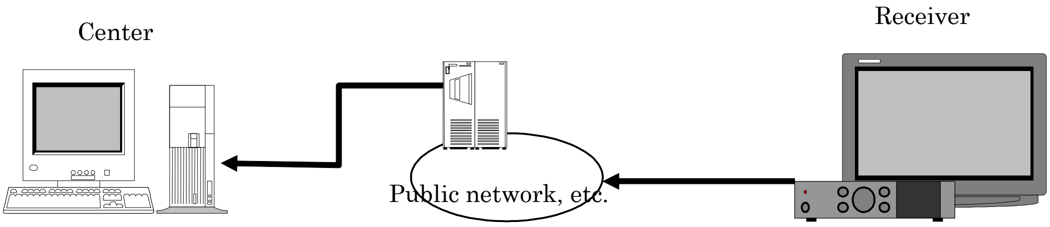


- Asymmetric bidirectional interactive connection models

4) Massive calls reception service: In this model, during data transmission between a receiver and a center, data is processed among the network in some ways (e.g. accumulating statistics). Actual data processing depends on a service. As a good example of a massive call reception service, which works with a broadcasting service, mass-calling services are available. In a typical massive call reception service, the number of incoming calls from receivers is counted with an incoming call exchange and the results are transmitted to a center in a sequential manner.

Advantages: A receiver can have simple implementation. A center benefits from less processing load arising from data aggregation or other data processing tasks.

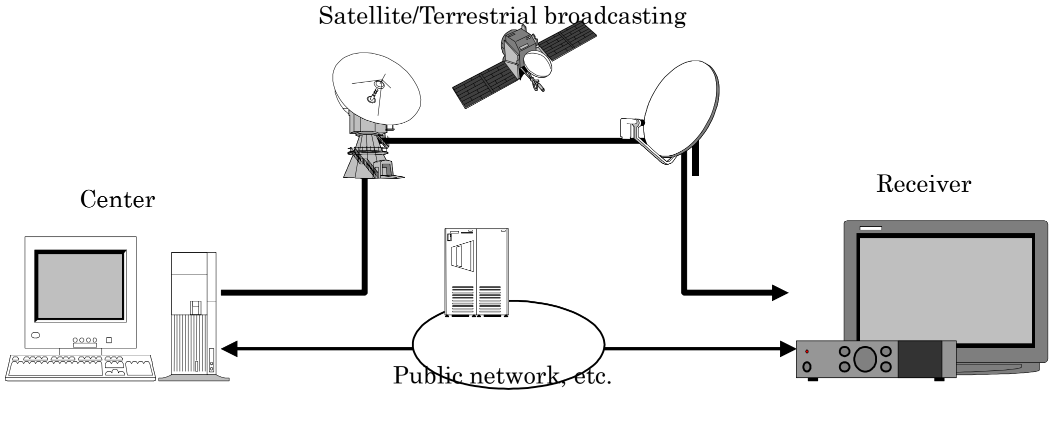
Disadvantages: Some massive calls reception services require subscription contracts with telecommunications operators.

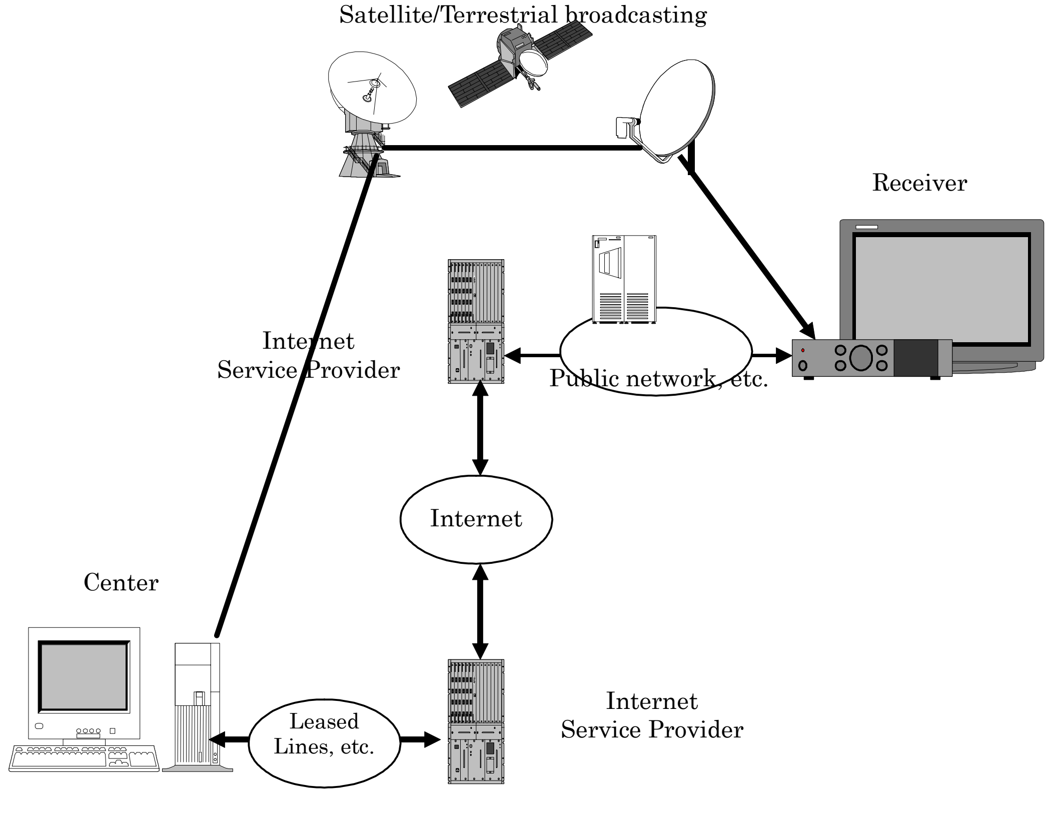


5) Public network (uplink) + broadcasting channel (downlink): In bidirectional communications, a public network carries uplink signals including requests and a broadcast channel carries responses to requests.

Advantages: When a broadcast channel via satellite or terrestrial is used for distribution of large volume of shared data, services can be provided at lower cost. This may lead to various new applications that would not be viable in the conventional broadcasting/telecommunication systems.
Receivers can communicate each other through an upstream channel, a downstream channel, and a center.

Disadvantages: A system supporting this type of connection is rather complicated. When additional protocols are needed to support a linkage of public networks as uplink and satellite/terrestrial broadcasting channels as downlink, extensive development initiatives may be required.





(2) Security levels and requirements

- Security is not assured:

  A message is transferred in a plaintext form.

- Security is assured to a degree depending on an implementation:

Encryption:	A message is encrypted to ensure that no third party, who is not an intended recipient, grasp the meaning of the message.
Data integrity:	A message is verified whether or not it arrives at an intended communicating party without being tampered nor transmission errors.
Authentication of a communicating party:	A communicating party verifies whether or not the other communicating party is an intended communicating party.
Digital signature:	A digital signature is designed to identify an originating party of a message and assure that the message has not been tampered nor compromised.

(3) Necessity of congestion control

Unlike transmissions over a conventional telephone network, transmissions in a service supported by an interactive channel and a broadcasting system are considered to link to a specific broadcasting program. This implies that some services may lead to resource-intensive massive transmissions in a rather short period of time, bring congestion to underlying networks. Once congestion has arisen in a network, normal operation of a broadcasting service is hindered by aborted transmissions from end users or other failures. These troubles, in turn, affect other communications on the network. This is why congestion must be prevented. To prevent congestion, how a service is operated, how a congestion control function is implemented in a receiver, or others must be well considered and designed in advance.

6 Cryptographic methods for interactive transmission systems

(1) Encryption-related algorithms

- Secret key cryptography

  Secret key cryptography algorithms are registered in JIS X5060.

  Note that the algorithms registered in JIS X5060 do not necessarily ensure their safety. Careful consideration is needed to decide which algorithm is to be used.

  Depending on a service, an algorithm employing a 128-bit secret key may be recommended. More specifically, the RC5 algorithm, that is in use on the Internet, and the Advanced Encryption Standard in the United States [C10] are good options. Among 64-bit block encryption algorithms, TripleDES [C11], RC5 [C12], MISTY, and FEAL32 which are registered in JISX 5060, which have been widely used, and whose encryption strengths have been proven, are recommended options.

- Public key cryptography

Evidence for security	Public Key Cryptography		Digital Signature	
	Algorithm	Remarks	Algorithm	Remarks
Prime factorization and others (not proven)	RSAES-EPOC	PKCS #1 Ver. 2 (July 1998)	RSASSA-PKCS1-v1_5	De facto standard
			Fiat-Shamir signature	A viable option for zero-knowledge and interactive signature.
			ESIGN	Characterized by the high speed of processing
Prime factorization and others (proven)	EPOC (hashed)	Eurocrypto '98	-	-
Discrete logarithm problem	Diffie-Hellman key distribution	A viable option for key distribution	DSA	NIST
	ElGamal	Crypto '84	Shnorr	-
	Cramer-Shoup	Crypto '98		
Elliptic curve discrete logarithm problem	Elliptic curve ElGamal	Characterized by the shorter key length	Elliptic curve DSA	-
			Elliptic curve Schnorr	-

The above table omits derivative/improved algorithms of each encryption algorithm.

(2) Key management

Key management includes key storage, key generation, key update, and key revocation. A single flaw in any area would lower the level of security. No area could be disregarded.

- Key storage:

The key storage concerns safety of a place to store secret keys for public key cryptography and secret key cryptography. Each safety level largely depends on the following items shown in the table below. This table summarizes security requirements, assuming that host sites and human resources are strictly managed in a center, and that a receiver is for home use and getting some attacks except massive attacks. To operate key storage procedures practically, these security requirements must be modified taking into account security policies.

As a typical storage operation of a secret key for public key cryptography and a master key for secret key cryptography, any key is encrypted with another key for secret key cryptography, instead of storing a readable value. To use the key, entering a PIN, a password, or other predefined information to use the key is required.

	Center	Receiver (end user)
Installed site	Can afford to be applied to a higher level of safety	Vulnerable to attacks
Access control	Can be strictly managed	Cannot be managed
Operator education/management	Can be strictly managed	Cannot be managed
Tamper registrant	Intermediate Can be reinforced with other items.	Most important item for a receiver. Cannot be reinforced with other items.

	Center	Receiver (end user)
Housing/enclosure	Requires reasonable consideration	Very important
Wiring circuit on a board	Requires reasonable consideration	Requires reasonable consideration in case of a vulnerable housing/enclosure
Terminals	Requires reasonable consideration	Requires reasonable consideration in case of a vulnerable housing/enclosure
LSI structure	Requires reasonable consideration	Requires reasonable consideration in case of a vulnerable housing/enclosure
Illegibility of software	Requires reasonable consideration	Requires reasonable consideration in case of a lower tamper resistance
Difficulty in analysing firmware program	Requires reasonable consideration	Requires reasonable consideration in case of a lower tamper resistance
Access control to memory	Requires reasonable consideration	Requires reasonable consideration in case of a lower tamper resistance

FIPS PUB 140-1 [C13] defines four security levels and provides security requirements for each level.

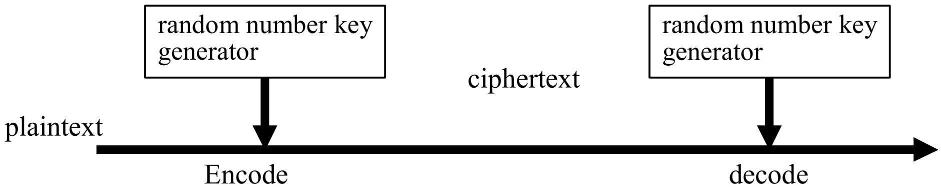
- Key generation/revocation:

A key used in a secret key cryptosystem is a random number, which is relatively easy to generate. In a public key cryptosystem, to generate a quality key, considerable efforts are devoted to program developing and computing. Some system configurations also require a center for generating keys or other facilities. An appendix of the X.509 document states how to generate a key for an RSA encryption algorithm. How to revoke a key is also an essential consideration to estimate validity of a digital signature. In most cases, a center must be responsible for monitoring current states of keys including how a key has been renewed and how a key has been revoked.

- Key renewal:

No encryption algorithm ensures permanent safety of a generated key. Any generated key requires to be renewed. In some public key cryptography implementations, the expiration period for a key is defined as some two years. In a case where a secret key cryptosystem works with a public key cryptosystem, most keys for secret key cryptosystem are used as session keys (one-time keys).

When only a public key cryptosystem is used as key cryptography, a hierarchical structure of key management must be employed. Any master key, that belongs to the highest level and is essential, must be used as little as possible to prevent exposure to risks.

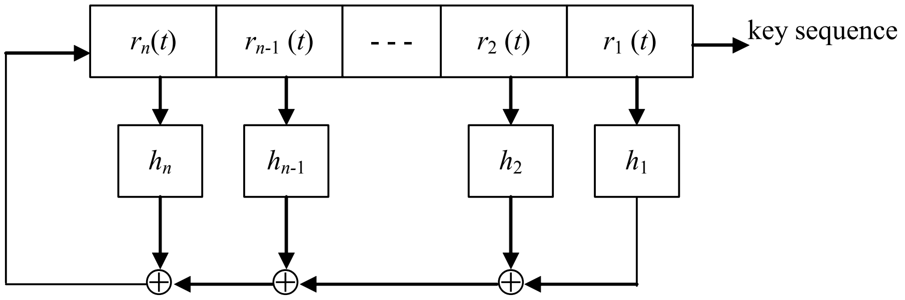


(3) Simple encryption

An example of a simple encryption method is a synchronized serial cryptosystem which is a Linear Feedback Shift Register mechanism, employing the Vernam cipher with the M-series random number generator. Since the linearity makes this algorithm vulnerable to known plaintext attacks, careful consideration must be given to operate this algorithm.

The Vernam cipher is a basic encryption algorithm, as simply illustrated in the diagram below.

Outputs from a Linear Feedback Shift Register are used as random number generators for a Vernam cipher implementation. An example using the M series is shown below.



$r_i(t)$: bit register value h_i: function for register value conversion

(4) Hash function

SHA-1 [C14], MD5 [C15], and MD2 [C16] have been widely used to operate a message digest algorithm.

(5) Expandability considerations of security components

To support higher performance in computing and wider distribution of multimedia data in future, security technologies continue to be developed and updated. Each security component is recommended to be expandable enough to employ newer technologies as required.

- Secret key cryptography

  As computing performance increases, 128-bit secret key cryptography algorithms are replacing existing 64-bit secret key cryptography algorithms. Recently, encryption algorithms that have provable security (it does not mean that the security is assured, but the level of security is provable) have been developed.

- Public key cryptography

  As computing performance increases, keys for public key cryptography continue to be longer in bits. Encryption algorithms with provable security and public key cryptography algorithms using elliptic curve functions are focused on in many research and development projects. Considering algorithm maturity and required encryption strength, currently used algorithms are recommended to be replaced in the future.

- Copyright management

  As multimedia data are distributed wider and digital data are vulnerable to unauthorized duplication, some types of contents need an appropriate copyright management. To cope with this challenge, electronic watermark technologies that embed copyright information within contents or other appropriate technologies must be employed.

Reference materials

(1) ARIB STD-B10 Ver. 4.0 "Service Information for Digital Broadcasting System" (2004, Dec.)

(2) ISO/IEC 13818-1 (2000) “Information Technology - Generic Coding of Moving Pictures and Associated Audio: SYSTEMS Recommendation H.220.0”

(3) ISO/IEC 13818-2 (1996) “Information Technology - Generic Coding of Moving Pictures and Associated Audio: VIDEO”

(4) ISO/IEC 13818-3 (1998) “Information Technology - Generic Coding of Moving Pictures and Associated Audio: Audio”

(5) ISO/IEC 13818-6 (1998) “Information Technology - Generic Coding of Moving Pictures and Associated Audio Information: Extensions for Digital Storage Media Command and Control”

(6) ISO/IEC 13818-7(1997) “Information Technology - Generic coding of moving pictures and associated audio information: Advanced Audio Coding (AAC)”

(7) ISO 639-2 (1996) “Codes for the representation of names of languages - Part 2: Alpha-3 code”

(8) ISO 8859-1 (1987) “Information processing - 8-bit single-byte coded graphic character sets - Part 1: Latin alphabet No.1”

(9) EN 301 192 (1997-12) “Digital Video Broadcasting (DVB); DVB Specification for Data Broadcasting”

(10) ATSC DVS-161 T3/S13 Doc.010 (1999) “Draft ATSC Standard "ATSC Data Broadcast Specification”

(11) DAVIC 1.4 Specification Part9 (1998) “Information Representation”

(12) The Notification No. 37 of Ministry of Internal Affairs and Communications in 2003

(13) The Notification No. 39 of Ministry of Internal Affairs and Communications in 2003

(14) The Notification No. 726 of Ministry of Internal Affairs and Communications in 2004

(15) RFC1521 (1993) Borenstein N., and N. Freed, “MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies” ,RFC 1521, Bellcore, Innosoft

---

(16) RFC1590 (1994) J.Postel, “Media Type Registration Procedure”, RFC 1590, ISI

(C1) ITU-T X.208 (1988-11) "Specification of abstraction construction describing format (ASN.1)"

Note: This Recommendation has been superseded by ITU-T X.680, X.681, X.682 and X.683.

(C2) ITU-T X.209 (1988-11) "Specification of basic encryption rule of abstraction construction describing format (ASN.1)"

Note: This Recommendation has been superseded by ITU-T X.690.

(C3) ITU-T H.234 (1994-11) "Encryption key management and authenticating system for audio visual service"

(C4) JIS X 5060 (1994) "Data encryption technology- Registration procedure of encryption algorithm"

(C5) JIS X 5055 (1996) "Security technology-Data completeness function using encryption inspection function by block encryption algorithm"

(C6) JIS X 5057-1 (1996) "Security technology - Hash function - Part 1: Introduction"

(C7) JIS X 5057-2 (1996) "Security technology - Hash function - Part 2: Hush function using n bit-block encryption algorithm"

(C8) JIS X 5056-3 (1996) "Security technology-- Entity authentication function - Part 3 - Authentication function using open key algorithm"

(C9) ITU-T, X.509 (1997-8) "Directory - Frame of authentication"

(C10) http://www.nist.gov/aes (1999-3) “Advanced Encryption Standard”

(C11) FIPS PUB 46-2, http://www.itl.nist.gov/div897/pubs/fip46-2.htm (1993-12) “DATA ENCRYPTION STANDARD (DES) “

(C12) RC5, RFC2040 (1996-10) “The RC5 Encryption Algorithm”

(C13) FIPS PUB 140-1, http://www-09.nist.gov/div897/pubs/fip140-1.htm (1994-1) “security requirements for cryptographic modules”

(C14) FIPS PUB 180-1, http://www.itl.nist.gov/div897/pubs/fip180-1.htm (1995-4) “SECURE HASH STANDARD”

(C15) MD5, RFC1321 (1992-4) ”The MD5 Message-Digest Algorithm”

(C16) MD2, RFC1319 (1992-4) “The MD2 Message-Digest Algorithm”

(C17) TLS1.0 ,RFC2246(1999-1) “The TLS Protocol Version 1.0”

DATA CODING AND TRANSMISSION SPECIFICATIONS
FOR DIGITAL BROADCASTING

ARIB STANDARD

ARIB STD-B24　VERSION 5.0-E1
VOLUME3
(May 29, 2006)

This Document is based on the ARIB standard of “Data Coding and Transmission Specifications for Digital Broadcasting” in Japanese edition and translated into English on December, 2006.

Published by

Association of Radio Industries and Businesses

Nittochi Bldg. 11F
1-4-1 Kasumigaseki, Chiyoda-ku, Tokyo 100-0013, Japan

TEL 81-3-5510-8590
FAX 81-3-3592-1103

Printed in Japan