ATSC-Mobile DTV Standard, Part 3 – Service Multiplex and Transport Subsystem Characteristics

Document A/153 Part 3:2009, 15 October 2009

Advanced Television Systems Committee, Inc.
1776 K Street, N.W., Suite 200
Washington, D.C. 20006

Mailing address	Advanced Television Systems Commmittee, Inc. 1776 K Street, N.W., Suite 200 Washington, D.C. 20006
Telephone	202-872-9160 (voice), 202-872-9161 (fax)
Web site	http://www.atsc.org, E-mail: standard@atsc.org

The revision history of this document is given below.

A/153 Revision History

A/153 approved	15 October 2009
Initial release of document	28 November 2009
Final publication of Part 3	1 April 2010
Editorial corrections made to Table 7.2 and Section 7.3	3 September 2010

Table of Contents

Index of Tables and Figures

ATSC Standard: ATSC Mobile DTV Standard, Part 3 – Service Multiplex and Transport Subsystem Characteristics

1 SCOPE

This Part describes the ATSC Mobile DTV (mobile/handheld, or simply M/H) system service multiplex and transport subsystem characteristics. Included are the following functions:

- Transport: adaptation between the RF transmission system and the remainder of the M/H system
- Signaling: metadata about content currently in the transport
- Streaming delivery: transport of streaming content
- File delivery: transport of files
- Interaction channel: characteristics of the interaction (return) channel

This standard was prepared by the Advanced Television Systems Committee (ATSC) Technology and Standards Group (TSG). It was approved by the full membership of the ATSC on 15 October 2009.

1.1 Organization

This document is organized as follows:

- **Section 1** – Outlines the scope of this Part and provides a general introduction.
- **Section 2** – Lists references and applicable documents.
- **Section 3** – Provides a definition of terms, acronyms, and abbreviations for this Part.
- **Section 4** – ATSC-M/H system overview
- **Section 5** –Transport-and signaling system overview
- **Section 6** – Transport system specifications
- **Section 7** – Service signaling system specifications
- **Section 8** – Time of day signaling
- **Section 9** – File delivery mechanism
- **Section 10** – Streaming delivery mechanism, including UDP encapsulation of real-time streaming data, RTP encapsulation of real-time streaming data, and a buffer model for real-time streaming data
- **Section 11** – Interaction channel
- **Annex A** – Error indicator in the M/H TP headers (informative)
- **Annex B** – Allocation of MH_service_id numbers (normative)

2 REFERENCES

At the time of publication, the editions indicated below were valid. All standards are subject to revision, and parties to agreement based on this standard are encouraged to investigate the possibility of applying the most recent editions of the documents listed below.

2.1 Normative References

The following documents contain provisions which, through reference in this text, constitute provisions of this standard.

[1] IEEE: "Use of the International Systems of Units (SI): The Modern Metric System," Doc. IEEE/ASTM SI 10-2002, Institute of Electrical and Electronics Engineers, New York, N.Y.

[2] OMA: "DRM Specification V2.0," Doc., OMA-DRM-DRM-V2_0, Open Mobile Alliance, San Diego, CA.

[3] ATSC: "Program and System Information Protocol for Terrestrial Broadcast and Cable," Doc. A/65:2009, Advanced Television Systems Committee, Washington D.C., 14 April 2009.

[4] IANA: IP Address Allocations, http://www.iana.org/numbers/ .

[5] IETF: STD05 (originally RFC 791), "Internet Protocol," Internet Engineering Task Force, Reston, VA, September 1981.

[6] USGPO: "Numerical designation of television channels," Code of Federal Regulations, Title 47, Chapter I, Part 73, Subpart E, Section 73.603 (Cite: 47CFR73.603), U.S. Government Printing Office, October 2007.

[7] ATSC: "ATSC Digital Audio Compression Standard (AC-3, E-AC-3), Revision B," Doc. A/52B, Advanced Television Systems Committee, Washington D.C., 14 June 2005.

[8] ATSC: "ATSC Digital Television Standard, Part 3 – Service Multiplex and Transport Subsystem Characteristics," Doc. A/53 Part 3:2009, Advanced Television Systems Committee, Washington, D.C., 7 August 2009.

[9] OMA: "Service Guide for Mobile Broadcast Services", Version 1.0, OMA-TS-BCAST_Service_Guide-V1_0, Open Mobile Alliance, URL: http://www.openmobilealliance.org/.

[10] ISO/IEC: ISO/IEC 14496-10 (ITU-T H.264), International Standard (2007), "Advanced video coding for generic audiovisual services," International Standards Organization, Geneva, Switzerland.

[11] ISO/IEC: ISO/IEC 639.2/B, "Code for the Representation of Names of Languages – Part 2: alpha-3 code, as maintained by the ISO 639/Joint Advisory Committee (ISO 639/JAC)," International Standards Organization, Geneva, Switzerland.

[12] IETF: "RTP Payload for Transport of Generic MPEG-4 Elementary Streams," Doc. RFC 3640, Internet Engineering Task Force, Reston, VA, November 2003.

[13] ISO/IEC 14496-3: 2005: "Information technology - Generic coding of moving picture and associated audio information - Part 3: Audio" with corrigenda 2, 4 and 5, and Amendment 2 (2006) with corrigenda 2 ("Audio Lossless Coding (ALS), new audio profiles and BSAC extensions"), International Standards Organization, Geneva, Switzerland.

[14] IETF: "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)," Doc. RFC 3890, Internet Engineering Task Force, Reston, VA, September 2004.

[15] IETF: "SDP: Session Description Protocol," Doc. RFC 4566, Internet Engineering Task Force, Reston, VA, July 2006.

[16] IETF: "FLUTE – File Delivery over Unidirectional Transport," Doc. RFC 3926, Internet Engineering Task Force, Reston, VA, October 2004.

[17] OMA: "Service and Content Protection for Mobile Broadcast Services," Open Mobile Alliance, OMA-TS-BCAST_SvcCntProtection-V1_0, available from http://www.openmobilealliance.org.

[18] ISO/IEC: ISO/IEC 13818-1: 2007 + Amendment #3, "Information technology – Generic coding of moving pictures and associated audio information: Systems," International Standards Organization, Geneva, Switzerland.

[19] ATSC: "Content Identification and Labeling for ATSC Transport," Doc. A/57B, Advanced Television Systems Committee, Washington, D.C., 26 May 2008.

[20] CEA: "U.S. and Canadian Rating Region Tables (RRT) and Content Advisory Descriptors for Transport of Content Advisory Information Using ATSC Program and System Information Protocol (PSIP)," Doc. ANSI/CEA-766-C, Consumer Electronics Association, Arlington, VA, April 2008.

[21] IETF: Internet-Draft draft-ietf-ntp-ntpv4-proto-11.txt, "Network Time Protocol Version 4 Protocol and Algorithms Specification," Internet Engineering Task Force, Reston, VA, 5 September 2008. [Editor's note: IETF work in process; RFC number to be assigned. For status information see
https://datatracker.ietf.org/doc/draft-ietf-ntp-ntpv4-proto/
for archival copy see
http://tools.ietf.org/id/draft-ietf-ntp-ntpv4-proto-11.txt]

[22] OMA: "File and Stream Distribution for Mobile Broadcast Services," Version 1.0, Doc. OMA-TS-BCAST_Distribution-V1_0, Open Mobile Alliance, available from http://www.openmobilealliance.org/.

[23] IETF: "RTP: A Transport Protocol for Real-Time Applications," Doc. RFC 3550, Internet Engineering Task Force, Reston, VA, July 2003.

[24] IETF: "User Datagram Protocol," STD06 (originally RFC 768), Internet Engineering Task Force, Reston, VA, 28 August 1980.

[25] IETF: "DCCP: Datagram Congestion Control Protocol," Doc. RFC 4340, Internet Engineering Task Force, Reston, VA, March 2006.

[26] IETF: "Internet Time Synchronization: the Network Time Protocol," Doc. RFC 1129, Internet Engineering Task Force, Reston, VA, October 1989.

[27] ATSC: "ATSC Interaction Channel Protocols," Doc. A/96, Advanced Television Systems Committee, Washington, D.C., 3 February 2004.

[28] IETF: "UTF-8, a transformation format of ISO 10646, F. Yergeau," Doc. RFC 3629. November 2003.

2.2 Informative References

[29] ATSC: "ATSC Digital Television Standard, Part 2 – RF/Transmission System Characteristics," Doc. A/53 Part 2:2007, Advanced Television Systems Committee, Washington, D.C., 3 January 2007.

[30] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 1 – Mobile/Handheld Digital Television System," Doc. A/153 Part 1:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

[31] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 2 – RF/Transmission System Characteristics," Doc. A/153 Part 2:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

[32] IETF: Internet-draft-mehta-rmt-flute-sdp-05.txt, "SDP Descriptors for FLUTE," Internet Engineering Task Force, Reston, VA. [For status information see
https://datatracker.ietf.org/doc/draft-mehta-rmt-flute-sdp/
for archival copy see
http://tools.ietf.org/id/draft-mehta-rmt-flute-sdp-05.txt]

[33] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 6 – Service Protection," Doc. A/153 Part 6:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

[34] IETF: "Asynchronous Layered Coding (ALC) Protocol Instantiation," Doc. RFC 3450, Internet Engineering Task Force, Reston, VA.

[35] IETF: "Layered Coding Transport (LCT) Building Block," Doc. RFC 3451, Internet Engineering Task Force, Reston, VA.

[36] IETF: "Forward Error Correction (FEC) Building Block," Doc. RFC 3452, Internet Engineering Task Force, Reston, VA.

[37] IANA Protocol Registry, Port Numbers, http://www.iana.org/assignments/port-numbers.

[38] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 7 – AVC and SVC Video System Characteristics," Doc. A/153 Part 7:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

[39] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 8 – HE AAC Audio System Characteristics," Doc. A/153 Part 8:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

[40] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 5 – Presentation Framework," Doc. A/153 Part 5:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

[41] IANA: Ether Types, http://www.iana.org/assignments/ethernet-numbers.

[42] IANA: RTP Payload Types for Standard Audio and Video Encodings, http://www.iana.org/assignments/rtp-parameters.

[43] IETF: "Internet Protocol, Version 6 (IPv6) Specification," Doc. RFC 1883, Internet Engineering Task Force, Reston, VA, December 1995.

[44] ATSC: "ATSC Mobile/Handheld Digital Television Standard, Part 4 – Announcement," Doc. A/153 Part 4:2009, Advanced Television Systems Committee, Washington, D.C., 15 October 2009.

3 DEFINITION OF TERMS

With respect to definition of terms, abbreviations, and units, the practice of the Institute of Electrical and Electronics Engineers (IEEE) as outlined in the Institute's published standards [1] shall be used. Where an abbreviation is not covered by IEEE practice or industry practice differs from IEEE practice, the abbreviation in question is described in Section 3.3 of this document.

3.1 Compliance Notation

As used in this document, "shall" denotes a mandatory provision of the standard. "Should" denotes a provision that is recommended but not mandatory. "May" denotes a feature whose presence does not preclude compliance, which may or may not be present at the option of the implementer.

3.2 Treatment of Syntactic Elements

This document contains symbolic references to syntactic elements used in the audio, video, and transport coding subsystems. These references are typographically distinguished by the use of a different font (e.g., restricted), may contain the underscore character (e.g., sequence_end_code) and may consist of character strings that are not English words (e.g., dynrng).

3.2.1 Reserved Fields

reserved — Fields in this document marked "reserved" are not to be assigned by the user, but are available for future use. Receiving devices are expected to disregard reserved fields for which no definition exists that is known to that unit. Each bit in the fields marked "reserved" is to be set to '1' until such time as it is defined and supported.

3.3 Acronyms and Abbreviation

The following acronyms and abbreviations are used within this standard.

ALC – Asynchronous Layered Coding

ATSC – Advanced Television Systems Committee

ATSC-M/H – ATSC Mobile and Handheld

AVC – Advanced Video Coding (ITU-T H.264)

bslbf – Bit string, left bit first

CIT-MH – Cell Information Table for ATSC-M/H

CRC – Cyclic Redundancy Check

DIMS – Dynamic Interactive Multimedia Service

DNS – Domain Name System

ESG – Electronic Service Guide

FDT – File Delivery Table

FEC – Forward Error Correction

FIC – Fast Information Channel

FLUTE – File Delivery over Unidirectional Transport (RFC 3926)

GAT – Guide Access Table

GAT-MH – Guide Access Table for ATSC-M/H

HTTP – Hypertext Transfer Protocol

IETF – Internet Engineering Task Force

IP – Internet Protocol

LCT – Layered Coding Transport

LTKM – Long Term Key Message

M/H – Mobile and Handheld

NoG – Number of M/H Groups per M/H Subframe for a designated Parade

NTP – Network Time Protocol

OMA-BCAST – Open Mobile Alliance Broadcast

PRC – Parade Repetition Cycle

RME – Rich Media Environment

RTP – Real-time Transport Protocol

SDP – Session Description Protocol

SG – (Electronic) Service Guide

signed int – signed integer

SLT-MH – Service Labeling Table for ATSC-M/H

SMT-MH – Service Map Table for ATSC-M/H

STKM – Short Term Key Message

SVC – Scalable Video Coding

TCP – Transmission Control Protocol

TNoG – Total Number of M/H Groups including all the M/H Groups belonging to all M/H Parades in one M/H Subframe

TSI – Transport Session Identifier

UDLR – UniDirectional Link Route

UDP – User Datagram Protocol

uimsbf – Unsigned integer, most significant bit first

3.4 Terms

The following terms are used within this Part.

Baseband Process – A process which includes all the necessary functions to extract data out of the physical signal of an M/H Broadcast and make the data available to the transport layer subsystem.

Baseband Processor – A module of an M/H receiver which performs the baseband process.

IP multicast stream – An IP stream in which the destination IP address is in the IP multicast address range.

IP stream – A sequence of IP datagrams with the same source IP address and the same destination IP address.

Local M/H Service – A Service which appears in one and only one MH Broadcast. Typically this is a Service created by a local broadcaster which will not be transmitted by another broadcast facility other than a repeater.

M/H Broadcast – The entire M/H portion of a physical transmission channel.

M/H Ensemble (or simply "Ensemble") – A collection of consecutive RS Frames with the same FEC codes, where each RS Frame encapsulates a collection of IP streams.

M/H Group – At the packet level, a collection of 118 consecutive MHE-encapsulated MPEG-2 transport packets delivering M/H Service data; also, the corresponding data segments after interleaving and trellis coding.

M/H Multiplex – A collection of M/H Ensembles in which the same IP protocol version is used for all the IP datagrams in the collection, and the IP addresses of the IP streams in the M/H Services in the Ensembles have been coordinated to avoid any IP address collisions. A single M/H Multiplex may include one or more M/H Ensembles.

M/H Parade (or simply "Parade) – A collection of M/H Groups that have the same M/H FEC parameters. Each M/H Parade carries one or two M/H Ensembles.

M/H Service – A package of packetized streams transmitted via an M/H Broadcast, which package is composed of a sequence of events which can be broadcast as part of a schedule.

M/H Service Signaling Channel – A single stream incorporated within each M/H Ensemble. The current version of the M/H SSC uses a IP multicast stream to deliver M/H Service Signaling tables that include IP-level M/H Service access information.

M/H Slot – A portion of an M/H Sub-Frame consisting of 156 consecutive MPEG-2 transport packets. An M/H-Slot may consist of all main packets or may consist of 118 M/H packets and 38 main packets. There are 16 M/H Slots per M/H Sub-Frame.

M/H Subframe – One fifth of an M/H Frame; each M/H Subframe is equal in size to 4 VSB data frames (8 VSB data fields).

M/H TP – The term "M/H Transport Packet (M/H TP)" is used to designate a row of an RS Frame payload with two bytes header included. Thus, each RS Frame payload is composed of 187 M/H TPs.

Media Service – A synchronized grouping of media component streams such that the display of one or many of the streams results in an entertainment or entertainment like presentation.

Media Component Stream – A single element of a Media Service. Examples of Media Component Streams could be a video or English audio stream.

MTU – The maximum sized datagram that can be transmitted through the next network.

Program – A collection of associated media streams that have a common timeline for a defined period. A program corresponds to the common industry usage of "television program."

Protected Content – Media Stream that is protected according to the requirements of A/153 Part 6.

Regional M/H Service – A Service which appears in two or more MH Broadcasts. Typically this is a Service transmitted by more than one broadcast facility

Rights Issuer – An entity that issues Rights Objects to OMA DRM Conformant Devices (as defined in OMA DRM [2]).

Rights Object – A collection of Permissions and other attributes which are linked to Protected Content (as defined in OMA DRM [2]).

RS Frame – Two-dimensional data frame through which an M/H Ensemble is RS CRC encoded. RS Frame is the output of M/H physical layer subsystem. One RS Frame contains 187 rows of N bytes each in its payload, where the value of N is determined by the transmission mode of M/H physical layer subsystem, and carries data for one M/H Ensemble.

UDP stream – A sequence of UDP/IP datagrams with the same destination IP address and the same destination UDP port number.

4 ATSC-M/H SYSTEM OVERVIEW

Please see ATSC A/153 Part 1 [30] for an overall description of the M/H system. The ATSC Mobile/Handheld service (M/H) shares the same RF channel as a standard ATSC broadcast service described in ATSC A/53 [29]. M/H is enabled by using a portion of the total available ~19.4 Mbps bandwidth and utilizing delivery over IP transport. The overall ATSC broadcast system including standard (TS Main) and M/H systems is illustrated in Figure 4.1.

This Part relates to the service multiplex and transport subsystem characteristics. The transport system specifications define the formats and protocols used for delivery of data in the M/H Broadcast stream. The signaling system specifications define the data formats and delivery parameters of the tables and other data structures used to signal the content being delivered in the M/H Broadcast stream.

4.1 File Delivery

This Part describes the protocol used to deliver files over an ATSC-M/H channel. This is the only method available for delivering files, and all files delivered over an ATSC-M/H channel will use this method. This Part does not describe the purpose of the files. For example, a broadcaster could deliver service protection files including security keys, video files, or audio files using this method. The purpose of the file must be described using other techniques. For example, audio and video files are typically described in the ESG.

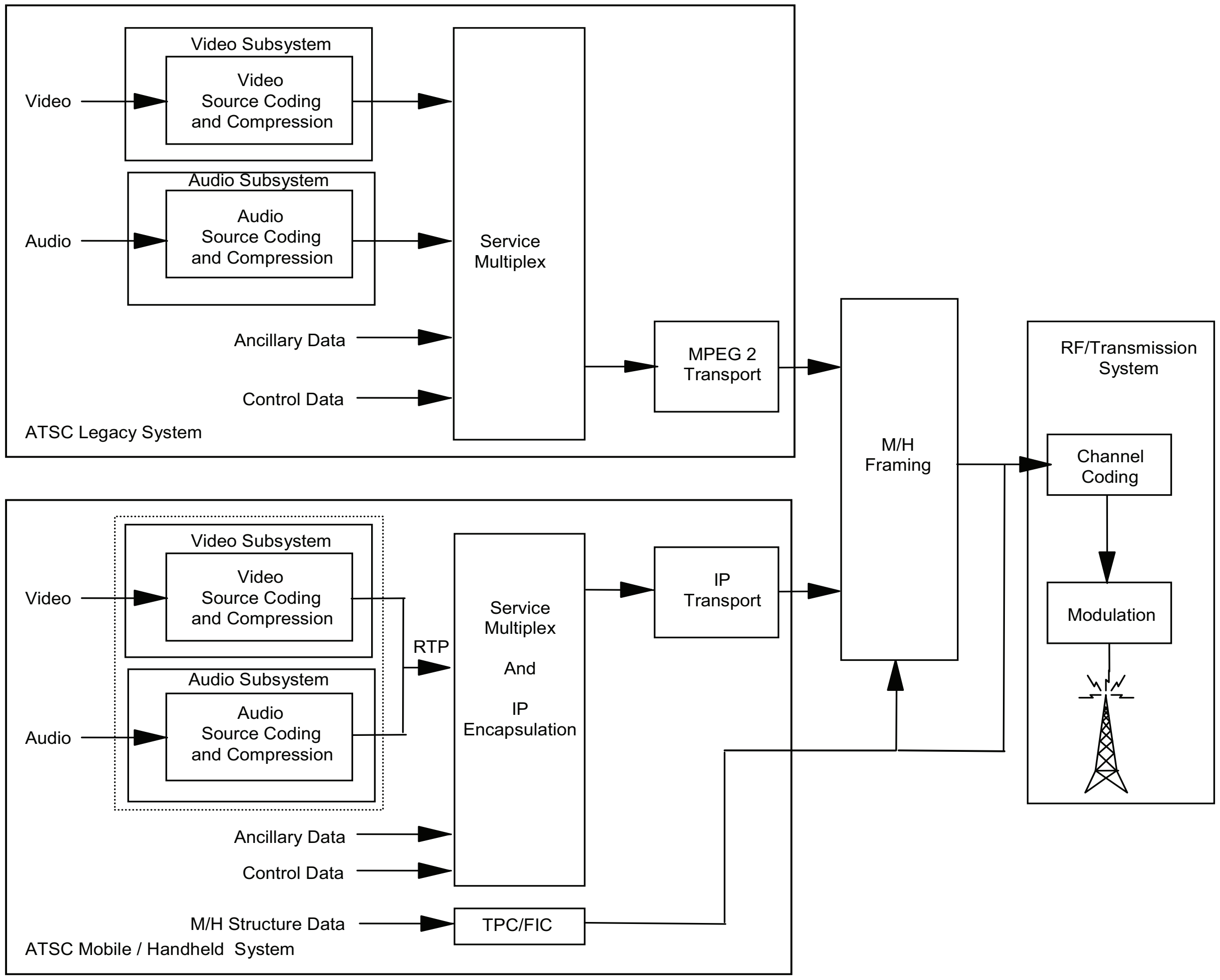

Figure 4.1 ATSC broadcast system with TS Main and M/H services.

4.2 Streaming Delivery

The ATSC-M/H system behaves very much like a standard data network delivery model. With this assumption, this Part specifies a standard for the Transport and Session Layer of the M/H digital television service when streaming real-time media. A classic OSI Reference Model is shown in Figure 4.2.

This explanatory mapping of the OSI Reference Model into the ATSC-M/H system uses Internet Protocol as the Network Layer. For the streaming of real-time media the Transport Layer of the ATSC-M/H system is defined to use User Datagram Protocol (UDP) and the Session Layer is defined to use Real-time Transport Protocol (RTP). Thus, when streaming real-time media the OSI Reference model when combined with ATSC-M/H components would look like the model shown in Figure 4.3.

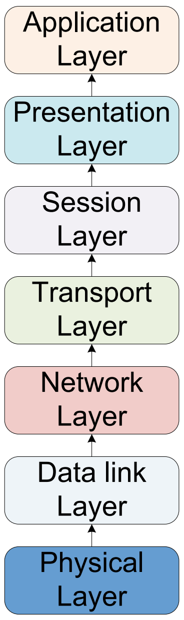

Figure 4.2 OSI reference model.

This Part also provides a streaming media buffer model. The goal of the buffer model is to allow for higher layer CODECs to have the information to enable a deterministically timed display of data as well as enforcing timing synchronization through the use of RTCP and NTP from service origination to reception equipment.

4.3 Interaction Channel

Remote interactivity requires the use of a two-way interaction channel that enables communications between the client device and remote servers. Examples of remote interactivity include E-commerce transactions during commercials, electronic banking, polling, email services, or other services yet to be defined.

5 TRANSPORT-AND SIGNALING SYSTEM

This section provides an overview of the ATSC-M/H Transport system and the ATSC-M/H Service Signaling system. For complete details on the RF/Transmission properties described in Sections 5.1, 5.2, and 5.3, see ATSC document A/153 Part 2 [31].

5.1 Time-Slicing Structure of ATSC-M/H Physical Layer Subsystem

In the ATSC-M/H physical layer system, the M/H data is transferred by a time-slicing mechanism to improve the receiver's power management capacity. Each M/H Frame time interval is divided into 5 sub-intervals of equal length, called M/H Subframes. Each M/H Subframe is in turn divided into 4 sub-divisions of length 48.4 msec, the time it takes to transmit one VSB frame. These VSB frame time intervals are in turn divided into 4 M/H Slots each (for a total of 16 M/H Slots in each M/H Subframe).

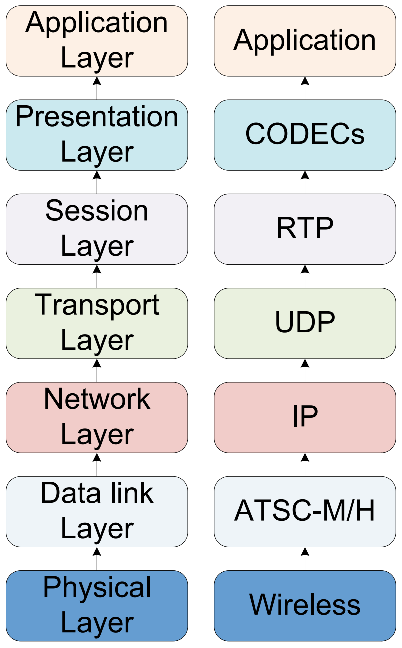

Figure 4.3 OSI reference model with ATSC-M/H components.

The M/H data to be transmitted is packaged into a set of consecutive RS Frames, where this set of RS Frames logically forms an M/H Ensemble. The data from each RS Frame to be transmitted during a single M/H Frame is split up into chunks called M/H Groups, and the M/H Groups are organized into M/H Parades, where an M/H Parade carries the M/H Groups from up to two RS Frames but not less than one. The number of M/H Groups belonging to an M/H Parade is always a multiple of 5, and the M/H Groups in the M/H Parade go into M/H Slots that are equally divided among the M/H Subframes of the M/H Frame as shown in Figure 5.1.

The actual algorithm for assigning M/H Groups to M/H Slots within each M/H Subframe is to number the M/H Groups from the first M/H Parade 1,…, n, and then the M/H Groups from the second M/H Parade n + 1,…, m, and then the M/H Groups from the third M/H Parade m + 1,…, k and so on until the M/H Groups are all numbered. Then they are assigned in order to slots 1, 5, 9, 13, 3, 7, 11, 15, 2, 6, 10, 14, 4, 8, 12, 16 (where the M/H Slots are numbered from left to right), for as many M/H Slots are needed. In the example in Figure 5.1, M/H Parade 0 is the first M/H Parade, and M/H Parade 1 is the second M/H Parade.

An M/H receiver is then able to turn on for only the M/H Slots that carry M/H Groups for the M/H Parades that carry the desired M/H Ensemble data. After every M/H Frame, the M/H receiver can gather the M/H Groups data received and form the desired RS Frames.

5.2 RS Frames

The RS Frame is the basic data delivery unit, into which the IP datagrams are encapsulated.

While an M/H Parade always carries a Primary RS Frame, it may carry an additional Secondary RS Frame as output of the physical layer subsystem. The number of RS Frames and the size of each RS Frame are determined by the transmission mode of the M/H physical layer

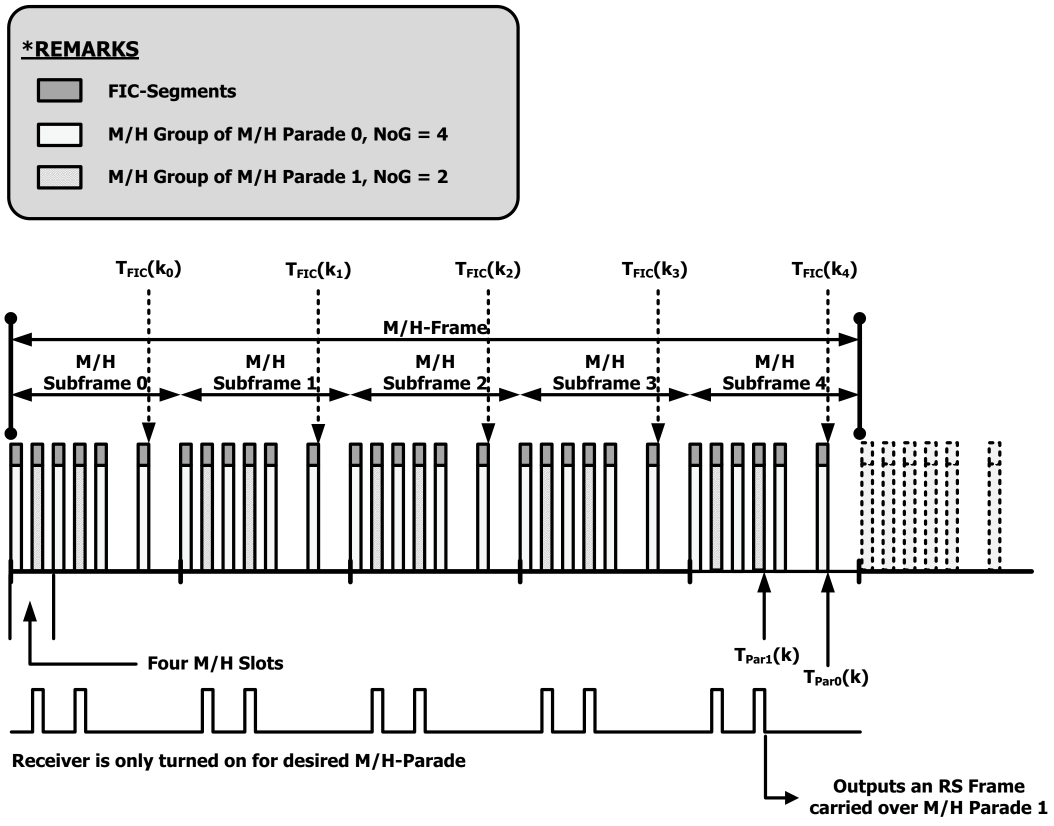

Figure 5.1 Time sliced structure of M/H physical layer system.

subsystem. Typically, the size of the Primary RS Frame is bigger than the size of Secondary RS Frame, when they are carried in one M/H Parade.

In order to retrieve and reconstruct the desired M/H Ensemble, an M/H receiver needs to have an RS Frame decoder, one for each RS Frame. Therefore, if an M/H receiver needs to process multiple RS Frames at the same time, then the receiver is required to have the same number of RS Frame decoders.

5.3 FIC: Fast Information Channel

The FIC is a separate data channel from the data channel delivered through RS Frames. The main purpose of the FIC is to efficiently deliver essential information for rapid M/H Service acquisition. This information primarily includes binding information between M/H Services and the M/H Ensembles carrying them so that the M/H receiver can select the proper M/H Ensemble that carries the M/H Service of interest.

The FIC delivers a data structure called an FIC-Chunk transmitted at least once within a single M/H Frame. The size of each FIC-Chunk is determined by the number of M/H Ensembles and the number of M/H Services provided in the M/H Broadcast. For its delivery, a single FIC-Chunk is divided into pieces that are packed into a number of delivery units, referred to as FIC-Segments. As many as 16 FIC-Segments are allowed in each FIC-Chunk. The size of each FIC-Segment is fixed at 37 bytes (2-byte Segment header and 35 bytes for payload). While the data carried through the RS Frame data channel is interleaved across an M/H Frame, the FIC-Segments, delivered through the FIC, are interleaved across each M/H Subframe. So de-interleaving an FIC-segment can proceed about 194 milliseconds after it begins to be received. Often, a complete FIC-Chunk will fit within the group of FIC-Segments delivered during a single M/H Subframe interval. This permits a receiver to access the binding information between M/H Services and the M/H Ensembles very rapidly via the FIC-Chunk, instead of having to evaluate each RS Frame delivered through the M/H Broadcast in order to determine which M/H Ensemble is carrying the M/H Service of interest. See Section 6.6 for further detail.

This standard also permits extended FIC-Chunks which span more than one sub-Frame within an M/H Frame. The FIC-Segments in such extended FIC-Chunks are interleaved on an M/H Subframe basis similarly to the FIC-Segments of FIC-Chunks that span just a single M/H sub-Frame.

5.4 M/H Service and M/H Ensemble

In ATSC-M/H, an “M/H Service” is similar in general concept to a virtual channel as defined in ATSC A/65 [3]. An M/H Service is a package of IP streams transmitted through M/H Multiplex, which forms a sequence of programs under the control of a broadcaster which can be broadcast as part of a schedule. Typical examples of M/H Services include TV services and audio services.

Collections of M/H Services are structured into M/H Ensembles, each of which consists of a set of consecutive RS Frames.

Normally, a single M/H Service is completely contained within a single M/H Ensemble. However, there may be situations where it is desirable to have an M/H Service that has components in multiple M/H Ensembles. A receiver must have the ability to decode multiple RS Frames concurrently in order to properly render such services, since a receiver needs to decode one sequence of RS Frames for each M/H Ensemble that it accesses concurrently.

It can facilitate rendering of services that span multiple Ensembles if all the IP datagrams in the ensembles have the same IP protocol version, and the UDP/IP addresses of the UDP/IP streams in the multiple Ensembles have been coordinated to avoid UDP/IP address collisions, so that a device can treat the Ensembles as a single IP subnet being accessed through a single network interface. A set of Ensembles in which the IP versions and UDP/IP addresses have been coordinated in this way is called an M/H Multiplex.

5.5 Hierarchical Signaling Architecture

Acquiring an M/H Service is done through two steps arranged in a hierarchical manner, namely access to the M/H Ensemble containing the M/H Service, and then access to the M/H Service. Two levels of signaling data for M/H Ensemble access and M/H Service access are defined.

- **For M/H Ensemble access**: The FIC-Chunk defined in Section 6.6.1, provides the binding information between M/H Service identifiers and M/H Ensembles, which is necessary for proper M/H Ensemble selection. Also, the FIC-Chunk provides some very basic information about each M/H Service and its placement in the transmission.
- **For IP level M/H Service access**: The Service Map Table (SMT-MH) defined in Section 7.3 provides IP address information for each M/H Service, all the IP stream component level information necessary for the M/H Service rendering, and other descriptive information about the Service.

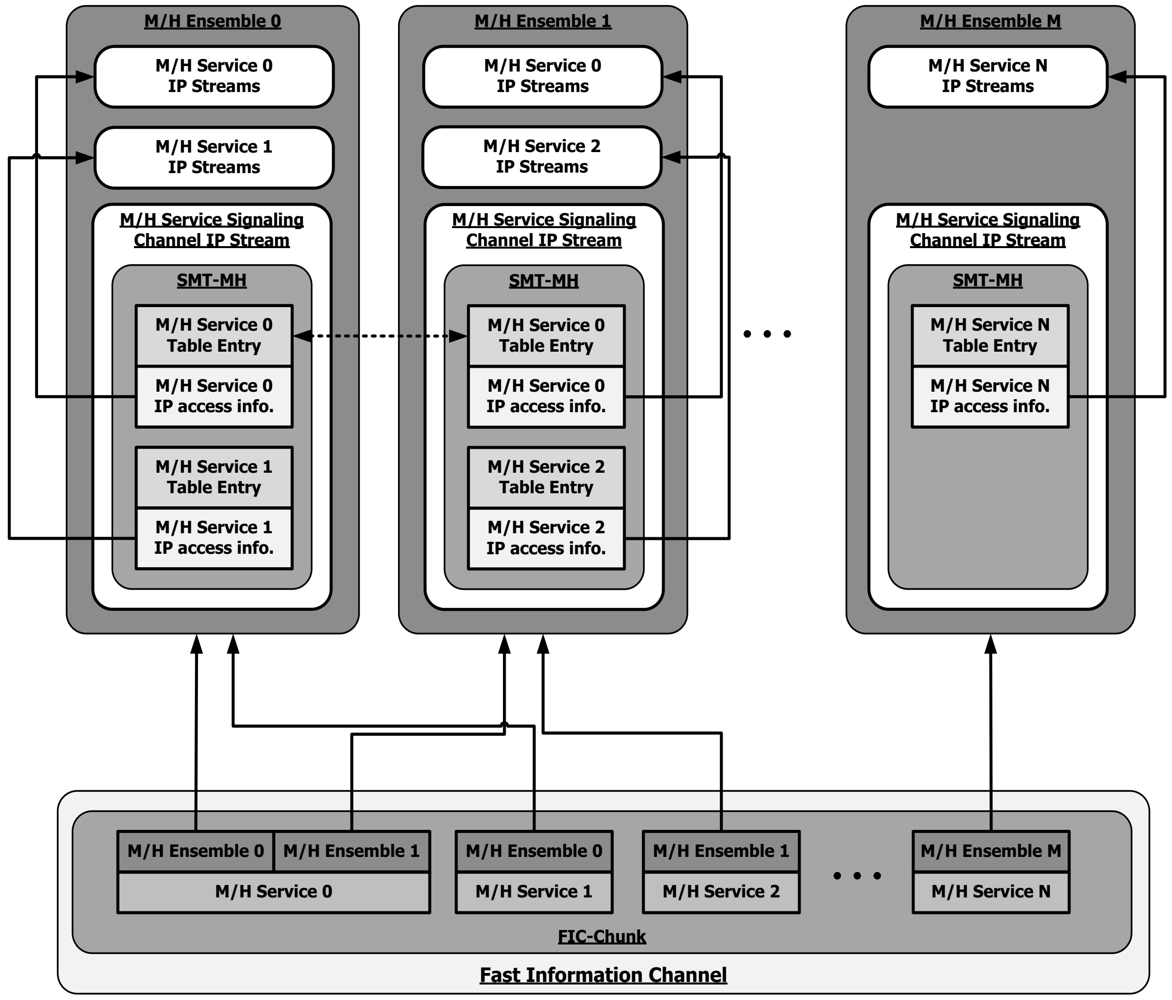
M/H Ensemble 0
M/H Service 0 IP Streams
M/H Service 1 IP Streams
M/H Service Signaling Channel IP Stream
SMT-MH
M/H Service 0 Table Entry
M/H Service 0 IP access info.
M/H Service 1 Table Entry
M/H Service 1 IP access info.
M/H Ensemble 1
M/H Service 0 IP Streams
M/H Service 2 IP Streams
M/H Service Signaling Channel IP Stream
SMT-MH
M/H Service 0 Table Entry
M/H Service 0 IP access info.
M/H Service 2 Table Entry
M/H Service 2 IP access info.
M/H Ensemble M
M/H Service N IP Streams
M/H Service Signaling Channel IP Stream
SMT-MH
M/H Service N Table Entry
M/H Service N IP access info.
M/H Ensemble 0
M/H Ensemble 1
M/H Service 0
M/H Ensemble 0
M/H Service 1
M/H Ensemble 1
M/H Service 2
M/H Ensemble M
M/H Service N
FIC-Chunk
Fast Information Channel

Figure 5.2 ATSC-M/H hierarchical signaling architecture.

Figure 5.2 illustrates the concept of the hierarchical signaling architecture applied to ATSC-M/H, including a situation when a single M/H Service spans across two M/H Ensembles, and those two M/H Ensembles form an M/H Multiplex.

6 M/H TRANSPORT SYSTEM SPECIFICATIONS (NORMATIVE)

This section defines the M/H Transport system, which provides an efficient way of carrying multimedia services over the ATSC M/H RF/Transmission system. The following specifications are included:

- Physical layer parameters
- Data encapsulation in RS Frames
- M/H services
- M/H multiplexes
- FIC signaling

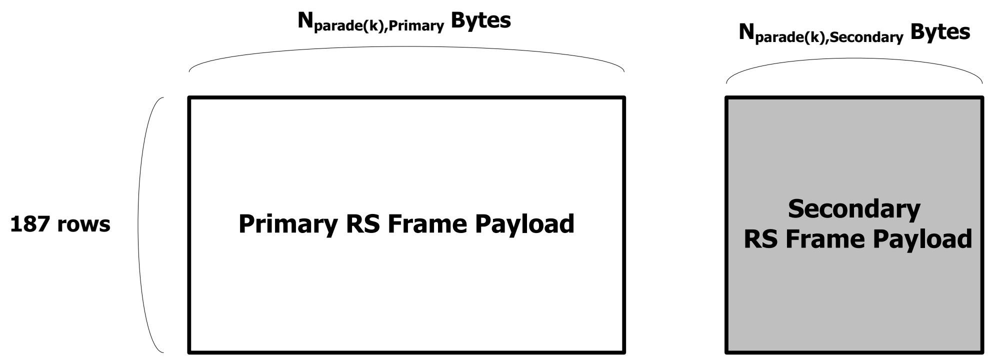
$N_{parade(k),Primary}$ Bytes
$N_{parade(k),Secondary}$ Bytes
187 rows
Primary RS Frame Payload
Secondary RS Frame Payload

Figure 6.1 Basic structure of the RS Frame payload.

6.1 Parameters Carried from M/H Physical Layer to M/H Transport

The M/H Transport system utilizes a number of parameters that are passed up to it from the M/H physical layer subsystem. The parameters listed below are transmitted repeatedly on every M/H Slot for which an M/H Group that belongs to any M/H Ensemble is delivered. Those parameters are:

- **M/H Ensemble ID:** Identification number for the M/H Ensemble which has a Group carried in this slot.
- **M/H Subframe number:** Identification number of the M/H Subframe that contains this slot.
- **TNoG:** Total Number of M/H Groups including all the M/H Groups belonging to all the M/H Parades in one M/H Subframe.
- **RS Frame Continuity Counter:** A number which serves as a continuity indicator of the RS Frames carried in the M/H Ensemble that contains this slot. The value begins at 0, modulo 16 and is incremented by 1 for each successive RF Frame.
- **N (Column size of RS Frame payload):** The column size of the RS Frame payload that belongs to the M/H Ensemble that contains this slot. This value determines the size of each M/H TP.
- **FIC Version Number:** A number that represents the version number of the FIC-Chunk carried in the M/H Subframe that contains this slot.

6.2 Data Encapsulation

An M/H Parade always carries a Primary RS Frame within each M/H Frame interval, and it sometimes carries a Secondary RS Frame as well, depending on the transmission configuration. Each RS Frame carries data for one M/H Ensemble in its payload. Figure 6.1 describes the basic structure of the RS Frame payload.

The size of the Primary RS Frame payload is 187 x $N_{parade(k),Primary}$ bytes, where the number of columns $N_{parade(k),Primary}$ of the RS Frame payload is the length of each row of the k-th M/H Parade's Primary RS frame. This value is determined by the transmission mode and is derived from parameters that are carried from the M/H physical layer subsystem. The size of the Secondary RS frame payload is 187 x $N_{parade(k),Secondary}$ bytes, where $N_{parade(k),Secondary}$ represents the length of each row of the k-th M/H Parade's Secondary RS Frame payload. This is also determined by the transmission mode and is also derived from parameters that are carried

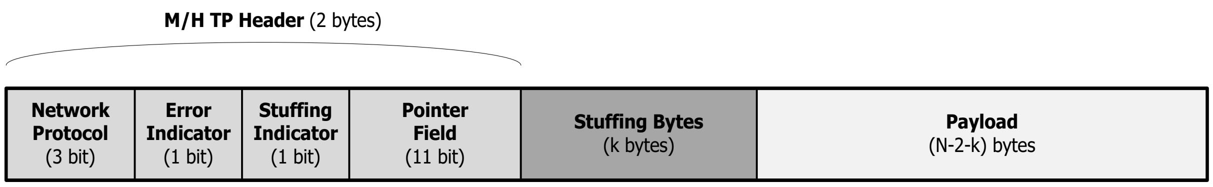

Figure 6.2 M/H TP format.

from the M/H physical layer subsystem. Typically, the size of Primary RS Frame is bigger than the size of Secondary RS Frame.

The term “M/H Transport Packet” (M/H TP) is used to designate a row of an RS Frame payload. Thus, each RS Frame payload is composed of 187 M/H TPs.

Each M/H TP shall contain a 2-byte header, leaving the rest of the row available for data, in the form of network protocol packets (e.g., IP datagrams).

The format of the two byte M/H TP header shall be as indicated in Figure 6.2. If the entire row is not used for payload, then stuffing bytes shall be inserted as shown in Figure 6.3.

The semantics of each field of an M/H TP are as follow:

network_protocol – This 3-bit field shall indicate the network protocol type of all the packets in the payload field of this row, as defined in Table 6.1.

Table 6.1 network_protocol Values

network_protocol Value	Meaning
000	IPv4 (conforming to RFC STD05 [5])
001-110	ATSC reserved
111	framed_packet_type, as defined in Table 6.3

error_indicator – This 1-bit indicator shall indicate whether any error was detected in this M/H TP. Such errors are detected on the decoder side by verifying a 16-bit CRC included with the M/H TP during encoding. See Section 5.4.2.1.2, “RS-CRC Encoder” of A/153 Part 2 [31] for a specification of this CRC. The value ‘0’ shall indicate no error was detected. The value ‘1’ shall indicate that an error was detected and the M/H TP should be discarded. The precise process for deriving the value of this error bit is described in Annex A.

Note: There is a small chance that an M/H TP with error_indicator field set to ‘1’ actually does not include any error. However, discarding the M/H TPs which have error_indicator field set to ‘1’ is recommended.

stuffing_indicator – This 1-bit indicator indicates whether any stuffing bytes are included in this M/H TP. The value ‘0’ shall indicate that there are no stuffing bytes and no length field, and the value ‘1’ shall indicate there is a length field and stuffing bytes prior to the payload.

pointer_field – This 11-bit field shall signal the starting point of the first new network protocol packet in the payload of this M/H TP. If this field is set to the maximum value, 0x7FF, then it shall mean there is no new network protocol packet starting in this M/H TP. If this field is set

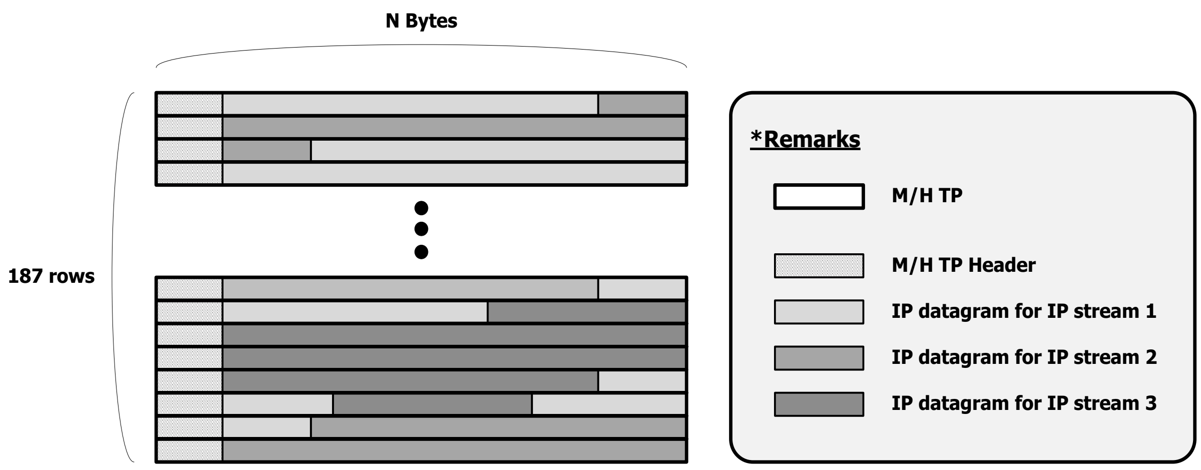
N Bytes
187 rows
*Remarks
M/H TP
M/H TP Header
IP datagram for IP stream 1
IP datagram for IP stream 2
IP datagram for IP stream 3

Figure 6.3 M/H TPs in the RS Frame payload.

to any other value, the value shall give the offset (in bytes) from the end of the header to the first byte of the first new network protocol packet that starts in the payload of this M/H TP.

stuffing_bytes – Stuffing bytes shall be present when stuffing_indicator =1. These are used to pad out a TP in case there are insufficient data available at the time of the framing. The stuffing_bytes field consists of a variable-size "length" sub-field followed by the actual stuffing bytes. The size of the "length" sub-field and its value shall be as defined in Table 6.2. Stuffing bytes (other than the length sub-field) shall be set to 0xFF.

payload() – This field contains the packet payload according to Table 6.1.

Table 6.2 Stuffing Bytes Values

Total Length N of Stuffing Bytes Field	Size of Length Sub-Field in Bytes	Value of Length Sub-Field
1	1	0xFF (and no stuffing bytes follow)
2	2	0xFEFF (and no stuffing bytes follow)
3-65278	2	N (and N-2 stuffing bytes follow)

Table 6.3 defines the framed_packet_type referenced in Table 6.1. This packet type is intended to accommodate possible future protocols for which the network packets do not contain a length field within the packets themselves.

Table 6.3 Framed Packet Syntax

Syntax	No. of Bits	Format
framed_packet() {		
ethernet_type	16	uimsbf
length	16	'11'
packet()	var	bslbf
}		

ethernet_type – This 16-bit field shall identify the type of packet in the packet() field according to the IANA registry [41]. Only registered values shall be used.

length – This 16-bit field shall be set to the total length in bytes of the packet() structure.

packet() – This variable length field shall contain a network packet.

The data in the rows shall consist of network protocol packets (e.g., IP datagrams), packed into the rows end-to-end, with possible wrapping around at the end of each row to the next row, as illustrated in Figure 6.3. If the RS Frame payload contains rows of multiple different network protocol types, then when a network protocol packet runs off the end of a row, it wraps around to the next row of the same network protocol type. If a network protocol packet runs off the end of a row, and there are no more rows in the RS Frame payload of the same network protocol type, then it wraps around to the next row in succeeding RS Frame payloads that has the same network protocol type.

6.3 IP Datagram Transport

The following constraints are applied to the IP streams and UDP streams delivered through an M/H Broadcast.

- The value of MTU shall be 1500 bytes[1].
- All IPv4 destination addresses used in Local M/H Services (which does not include M/H Service Signaling) shall be in the administrative multicast range of 239.0.0.0 to 239.255.255.255. The source address range is unconstrained.
- All IPv4 destination addresses used in Regional M/H Services (which does not include M/H Service Signaling) shall be in the range of 234.0.0.0 to 238.255.255.255. The source address for these datagrams must be an address which is attributed in the global DNS database to the Service Provider of the Regional M/H Service.
- All the IP datagrams delivered through a single M/H Ensemble shall all be of the same version of the IP protocol.
- IP fragmentation, as specified in the IETF IPv4 RFC STD05 [5], shall not be applied to RTCP/UDP/IP datagrams or to NTP/UDP/IP datagrams.
- Service protection, as defined in A/153 Part 6 [33], shall not be applied to NTP datagrams.
- NTP/UDP/IP datagrams shall not be split across RS Frame boundaries; i.e., no NTP datagram shall have a portion of it delivered at the end of one RS Frame payload in an Ensemble and the rest of it delivered at the beginning of the next RS Frame payload in the Ensemble.

6.4 M/H Service

An M/H Service is defined to be a package of IP streams transmitted through an M/H Broadcast, which forms a sequence of programs under the control of a broadcaster which can be broadcast as part of a schedule.

Each M/H Service shall have an associated 16-bit M/H Service ID, where the M/H Service ID shall uniquely identify the M/H Service within the scope of the M/H Broadcast.

The IP datagrams of the IP streams transmitted through an M/H Broadcast for a single M/H Service shall all be of the same version of the IP protocol.

1. Note that this value may increase in the future.

6.5 M/H Multiplex

Usually an M/H Service consists of a set of IP streams that are all contained within a single M/H Ensemble. However, as was pointed out in Section 5.4, there might be situations where it is desirable to have an M/H Service that includes IP streams from multiple M/H Ensembles. For example, to make the most efficient use of bandwidth the basic components of an enhanced service could be delivered in an ensemble with a high level of error correction, and the enhancements to the service could be delivered in an ensemble with a lower level of error correction.

As pointed out in Section 5.2, a receiver needs the ability to decode a sequence of RS Frames for each M/H Ensemble that it accesses concurrently. Thus, for the example above the enhancements could only be received by a receiver that has the ability to decode two or more sequences of RS Frames concurrently.

In general, address conflicts between IP datagrams in different M/H Ensembles are not a serious problem, since the IP datagrams from the different M/H Ensembles are coming out of different RS Frame decoders, and they can therefore be treated as if they are coming from different subnets through different network interfaces.

However, it can be desirable to allow receivers that have the ability to decode multiple RS Frames concurrently to treat a set of M/H Ensembles as a single subnet being accessed through a single network interface. To make this possible it is necessary to ensure that there are no UDP/IP address conflicts among the UDP/IP datagrams in the services contained in the set of M/H Ensembles.

The statement that a set of M/H Services has no UDP/IP address conflicts shall mean that for all UDP/IP streams within each M/H Service, each combination of destination IP address and destination UDP port from any UDP/IP stream in the service shall be unique to that M/H Service.

An M/H Multiplex is defined as a set of one or more M/H Ensemble(s) for which the same protocol version is used for all the IP datagrams in the collection, and the set of all the M/H Services in the ensembles have no UDP/IP address conflicts.

To facilitate processing of services that have components in more than one ensemble, whenever a service has components in multiple ensembles the set of ensembles containing components of the service shall meet the definition of an M/H multiplex.

Figure 6.4 shows the OSI 7-layer stack model as it could be implemented for an M/H Multiplex by a receiver with the ability to decode concurrently at least as many sequences of RS Frames as there are ensembles in the M/H Multiplex.

> Note: It is recommended to assume that the receiver devices deployed in the initial M/H market have the ability to decode only a single sequence of RS Frames concurrently. Therefore, it is recommended to restrict each service to a single Ensemble for initial ATSC-M/H broadcasts.

6.6 Signaling Through the Fast Information Channel

This section defines the contents of the FIC Chunk carried in the Fast Information Channel, and the encapsulation used for delivering the FIC-Chunk in the FIC segments. As described in Section 5.3, the FIC is an independent channel from the RS Frame data channel delivering the IP streams that belong to an M/H Ensemble.

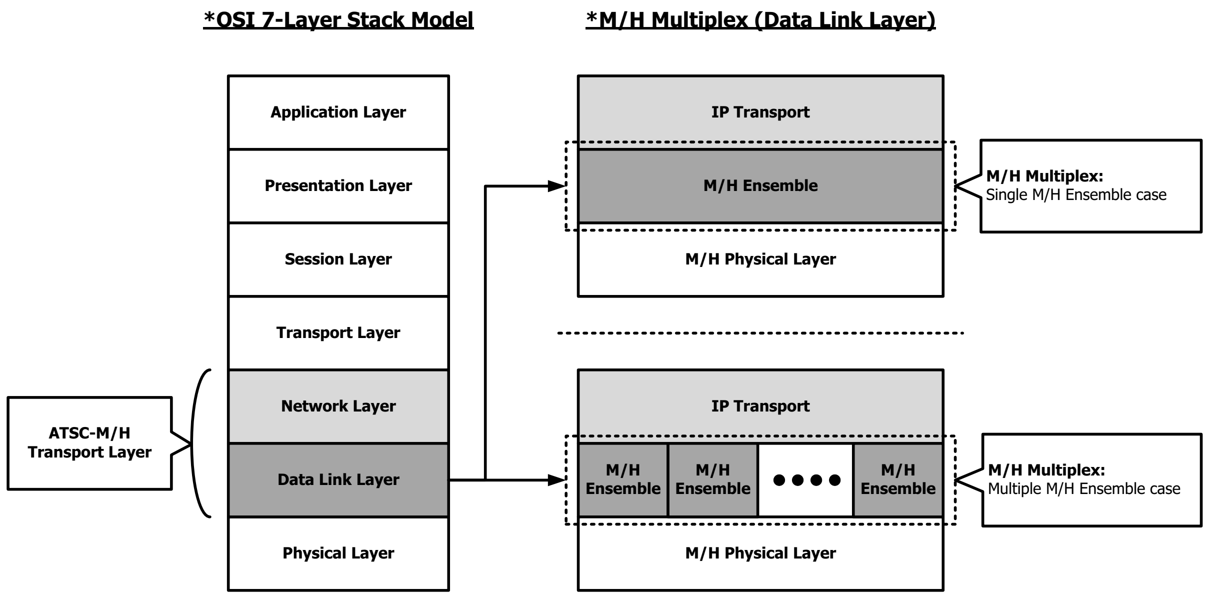

Figure 6.4 M/H Multiplex.

6.6.1 Contents of the FIC Chunk

The FIC-Chunk includes 5 bytes of FIC-Chunk header and a variable length payload. The total length of the payload is determined by the number of M/H Ensembles and the number of M/H Services provided in the M/H Broadcast. However, because of the way the FIC-Chunk is packed into the FIC-Segments, there is an upper limit of 560 bytes (35 bytes x 16, where 35 bytes is the fixed length of the payload of each FIC-Segment and 16 is the maximum number of FIC-Segments allowed for a single FIC-Chunk, including the FIC-Chunk header). This in effect imposes a limit (albeit a very high one—on the order of 140) on the number of M/H Services in the M/H Broadcast. The bit stream syntax for the FIC-Chunk shall be as shown in Table 6.4.

Table 6.4 Bit Stream Syntax for the FIC-Chunk

Syntax	No. of Bits	Format
FIC_chunk() {		
FIC_chunk_header()	5*8	
FIC_chunk_payload()	var	
}		

The bit stream syntax for the FIC-Chunk header shall be as shown in Table 6.5 and the bit stream syntax for the FIC-Chunk payload shall be as shown in Table 6.6, and the semantics for each field in those tables shall be as defined hereafter.

Table 6.5 Bit Stream Syntax for the FIC-Chunk Header

Syntax	No. of Bits	Format
FIC_chunk_header() {		
FIC_chunk_major_protocol_version	2	uimsbf
FIC_chunk_minor_protocol_version	3	uimsbf
FIC_chunk_header_extension_length	3	uimsbf
ensemble_loop_header_extension_length	3	uimsbf
MH_service_loop_extension_length	3	uimsbf
reserved	1	'1'
current_next_indicator	1	bslbf
transport_stream_id	16	uimsbf
num_ensembles	8	uimsbf
}		

FIC_chunk_major_protocol_version – A two-bit unsigned integer field that represents the major version of the syntax and semantics of the FIC-Chunk. A change in the major version level shall indicate a non-backward-compatible level of change. To conform to the current version of this standard, the value of this field shall be set to '00'. The value of this field shall be incremented by one each time the structure of the FIC-Chunk is changed in a non-backward-compatible manner from a previous major version of the FIC-Chunk, by a future version of this standard.

FIC_chunk_minor_protocol_version – A 3-bit unsigned integer field that represents the minor version of the syntax and semantics of the FIC-Chunk. A change in the minor version level, provided the FIC_chunk_major_protocol_version remains the same, shall indicate a backward-compatible level of change. To conform to the current version of this standard, the value of this field shall be set to '000'. The value of this field shall be incremented by one each time the structure of the FIC-Chunk is changed in backward-compatible manner from a previous minor version of the FIC-Chunk with the same major version, by a future version of this standard.

FIC_chunk_header_extension_length – A 3-bit unsigned integer field that represents the length of the extension field(s) of the FIC_chunk_header() added by one or more minor version level changes of the FIC-Chunk syntax. The value of this field shall indicate the total length in bytes of extension field(s) added by all minor version changes up to the current one (for the same major version). Such extension fields shall immediately precede the num_ensembles field of the FIC_chunk_header(), with fields added by higher minor protocol versions appearing after fields added by lower minor protocol versions (for the same major protocol version). The 3-bit length of this field requires that any change of syntax of the FIC-Chunk header which would push the total length of the extension(s) over 7 bytes will need to be treated as a major version change. To conform to the current version of this standard, the value of this field shall be set to '000'.

ensemble_loop_header_extension_length – A 3-bit unsigned integer field that represents the length of the extension field(s) of the header of the num_ensemble loop in the FIC_chunk_payload() added by one or more minor version level changes of the FIC-Chunk syntax. The value of this field shall indicate the total length in bytes of extension field(s) added by all minor version changes up to the current one (for the same major version). Such extension fields shall immediately precede the num_MH_services field of the FIC_chunk_payload(), with fields added by higher minor protocol versions appearing after fields added by lower minor protocol versions (for the same major protocol version). The 3-bit allocation of this field requires that any change of syntax of the num_ensembles loop header which would push the total length of the extension(s) over 7 bytes will need to be treated as a major version level of change. To conform to the current version of this standard, the value of this field shall be set to '000'.

MH_service_loop_extension_length – A 3-bit unsigned integer field that represents the length of the extension field(s) of the num_MH_services loop entry in the FIC_chunk_payload() added by one or more minor version level changes of the FIC-Chunk syntax. The value of this field shall indicate the total length in bytes of extension field(s) added by all minor version changes up to the current one (for the same major version). Such extension fields shall immediately follow the SP_indicator field of the FIC_chunk_payload(), with fields added by higher minor protocol versions appearing after fields added by lower minor protocol versions (for the same major protocol version). The 3-bit allocation of this field requires that any change of syntax of the num_MH_services loop entry which would push the total length of the extension(s) over 7 bytes will need to be treated as a major version level of change. To conform to the current version of this standard, the value of this field shall be set to '000'.

current_next_indicator – A one-bit indicator, which when set to '1' shall indicate that this FIC-Chunk is applicable to the M/H Frame in which it appears. When the bit is set to '0', it shall indicate that this FIC-Chunk will be applicable for the next M/H Frame.

transport_stream_id – This 16-bit unsigned integer number field serves as a label to identify this M/H Broadcast. The value of this field shall be equal to the value of the transport_stream_id field in the Program Association Table (PAT) in the MPEG-2 transport stream of the main ATSC broadcast.

num_ensembles – An 8-bit unsigned integer field that shall equal the number of M/H Ensembles carried through this physical transmission channel, including those that have a Parade Repetition Count greater than 0 and do not have any groups in the M/H Frame to which this FIC Chunk refers.

Table 6.6 Bit Stream Syntax for the FIC-Chunk Payload

Syntax	No. of Bits	Format
FIC_chunk_payload() {		
for (i=0; i<num_ensembles; i++) {		
ensemble_id	8	uimsbf
reserved	3	‘111’
ensemble_protocol_version	5	uimsbf
SLT_ensemble_indicator	1	bslbf
GAT_ensemble_indicator	1	bslbf
reserved	1	’1’
MH_service_signaling_channel_version	5	uimsbf
num_MH_services	8	uimsbf
for (j=0; j<num_MH_services; j++) {		
MH_service_id	16	uimsbf
reserved	3	‘111’
multi_ensemble_service	2	uimsbf
MH_service_status	2	uimsbf
SP_indicator	1	bslbf
}		
}		
FIC_chunk_stuffing()	var	
}		

ensemble_id – This 8-bit unsigned integer field identifies the associated M/H Ensemble. The value of this field shall be derived from the parade_id carried through the TPC (Transmission Parameter Channel), by using the parade_id of the associated M/H Parade for the least significant 7 bits, and using ‘0’ for the most significant bit when the M/H Ensemble is carried over the Primary RS Frames and using ‘1’ for the most significant bit when the M/H Ensemble is carried over the Secondary RS Frames. For further details about the TPC, see ATSC A/153 Part 2 [31]. Note that the value of ensemble_id of an M/H Ensemble shall not be changed during the period of time where an M/H Service is present and/or announced in the SG.

ensemble_protocol_version – A 5-bit unsigned integer field that represents the version of the structure of this ensemble, specifically its RS Frame payload structure and its M/H Service Signaling Channel structure. The value of this field shall be set as specified in Table 6.7.

Table 6.7 ensemble_protocol_version Values

Version	
0	The M/H ensemble configuration (the RS Frame payload structure and the M/H Service Signaling Channel configuration) that is defined in this version of this standard
1-31	Reserved for other M/H ensemble configurations possibly defined in future versions of this standard

Informative Note: A new value of the ensemble_protocol_version would be triggered by changes to its RS Frame payload structure and/or its M/H Service Signaling Channel structure that cannot be signaled by the signaling mechanisms within the RS Frame payload and the M/H Service Signaling Channel. Examples of the general scope of such possible changes include:

- A fundamental change in the way data are packed into the RS Frame payload, such as inserting data column by column instead of row by row, or defining an M/H Transport Packet to consist of two rows rather than one row (to cut TP header overhead in half).
- A change in the TP header and perhaps TP packet format, such as adding a TP protocol version field in the TP header, or changing the way stuffing bytes are signaled.
- A change in the IP address and port used for the Service Signaling Channel, perhaps even putting different tables at multiple different IP addresses or ports.
- Replacing the current signaling tables (SMT, GAT, etc.) with new versions that do not use MPEG-2 private section syntax, but have some totally different structure such as XML.

SLT_ensemble_indicator – A one-bit indicator, which when set to '1' shall indicate that the SLT-MH (Service Labeling Table) is carried in the M/H Service Signaling Channel of this M/H Ensemble. (See Sections 7.2 and 7.6 of this Part for the definition of the M/H Service Signaling Channel of an M/H Ensemble and the definition of the SLT-MH.)

GAT_ensemble_indicator – A one-bit indicator, which when set to '1' shall indicate that the GAT-MH (Guide Access Table) is carried in the signaling stream of this ensemble. (See Sections 7.2 and 7.4 of this Part for the definition the M/H Service Signaling Channel of an M/H Ensemble and the definition of the GAT-MH.)

MH_service_signaling_channel_version – This 5-bit field is the version number of the M/H Service Signaling Channel of this M/H Ensemble. The value of this field is modulo 32 and shall be incremented by 1, whenever a change is made in any of the tables carried within the M/H Service Signaling Channel in this ensemble.

num_MH_services – An 8-bit unsigned integer field that represents the number of M/H Services carried through this M/H Ensemble.

MH_service_id – A 16-bit unsigned integer number that identifies the M/H Service. This number shall be unique within the M/H Broadcast. See Annex B for a description of a scheme for specifying a local range and regional range for MH_service_id numbers. For a situation when an M/H Service has components in multiple M/H Ensembles, the set of IP streams of the service in each Ensemble shall be treated as a separate service for signaling purposes, except that the entries for these services in the FIC shall all have the same MH_service_id value. Thus, the same MH_service_id value may appear in more than one num_ensembles loop, and when this happens the MH_service_id shall represent the overall combined service, thereby maintaining the uniqueness of the MH_service_id.

Note: For any situation when a receiver presents the value of the MH_service_id to a viewer, it is recommended that the value of MH_service_id to be presented in two parts (the higher 8 bits forming the first number and the lower 8 bits forming the last number) separated by an appropriate delimiter.

multi_ensemble_service – A two-bit enumerated field that shall identify whether this M/H Service is carried across more than one M/H Ensemble. Also, this field identifies whether the M/H Service can be rendered meaningfully with only the portion of the M/H Service carried through this M/H Ensemble. The value of this field is defined in Table 6.8.

Table 6.8 Multi Ensemble Service

multi_ensemble_ service	Meaning
‘00’	All the IP streams that form this M/H Service are delivered through this M/H Ensemble.
‘01’	The IP streams that form this M/H Service are delivered through multiple M/H Ensembles. A meaningful version of this M/H Service can be rendered with only IP stream components delivered through this M/H Ensemble.
‘10’	The IP streams that form this M/H Service are delivered through multiple M/H Ensembles. No meaningful version of this M/H Service can be rendered with only the IP stream components delivered through this M/H Ensemble.
‘11’	[Reserved for future ATSC use]

MH_service_status – A 2-bit enumerated field that shall identify the status of this M/H Service. The most significant bit indicates whether this M/H Service is active (when set to 1) or inactive (when set to 0) and the least significant bit indicates whether this M/H Service is hidden (when set to 1) or not (when set to 0).

SP_indicator – A 1-bit field that indicates, when set, service protection is applied to at least one of the components needed to provide a meaningful presentation of this M/H Service.

FIC_chunk_stuffing – Stuffing may exist in an FIC-Chunk. The number of stuffing bytes shall be the minimum number needed to make the length of the FIC-Chunk evenly divisible by 35.

6.6.2 FIC-Chunk Segmentation and Delivery

A 37-byte signaling data delivery unit called an FIC-Segment shall be sent in each M/H Group. Thus, within an M/H Frame, the total number of FIC-Segments is equal to (TNoG x 5), which provides (37 x TNoG x 5) bytes for FIC-Chunk delivery. Each FIC-Segment has a two-byte header and 35 bytes for payload.

Each FIC-Chunk instance to be transmitted shall be divided into a number of delivery units of size 35 bytes in length. The number of delivery units is determined by the size of the FIC-Chunk instance, which in turn is determined by the number of M/H Ensembles and the number of M/H Services present in the M/H Broadcast. Each delivery unit shall be packed into an FIC-Segment for transmission. Figure 6.5 illustrates the segmentation of the FIC-Chunk.

The following restrictions and recommendations apply to the delivery of the FIC-Chunks through FIC-Segments:

- The length of an FIC-Segment shall be equal to 37 bytes, consisting of a 2-byte header and a 35 byte payload. The syntax of the header shall conform to Table 6.9, and the semantics of the header fields shall conform to the definitions following Table 6.9.

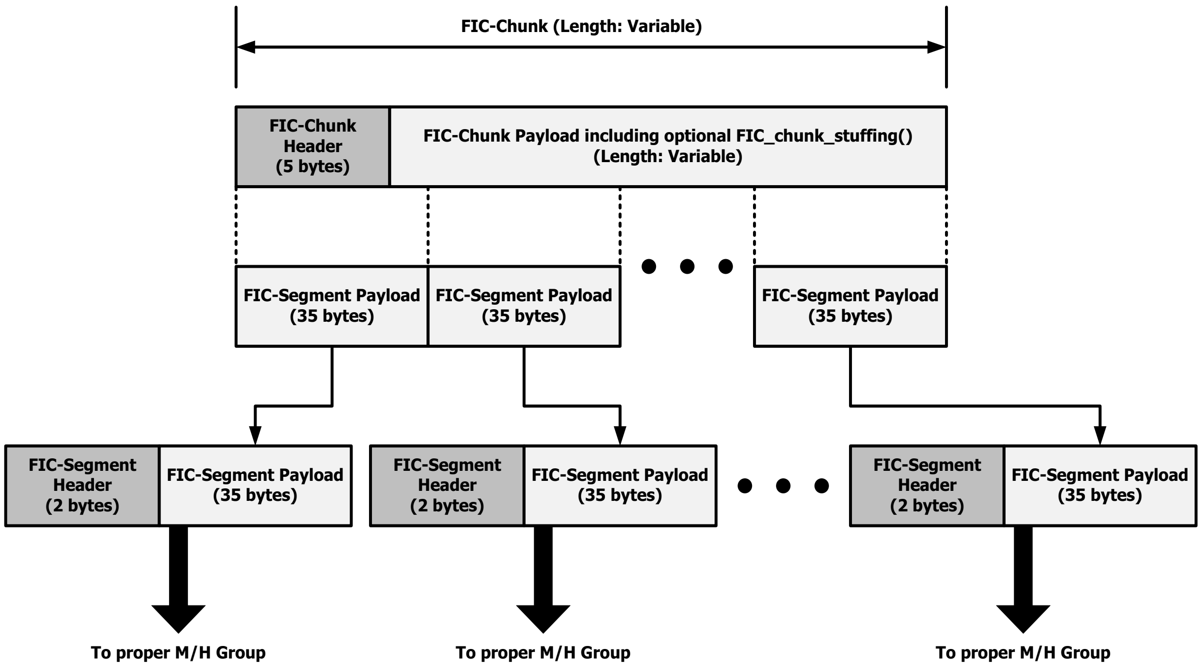
FIC-Chunk (Length: Variable)
FIC-Chunk Header (5 bytes)
FIC-Chunk Payload including optional FIC_chunk_stuffing() (Length: Variable)
FIC-Segment Payload (35 bytes)
FIC-Segment Header (2 bytes)
To proper M/H Group

Figure 6.5 FIC-Chunk segmentation.

- An FIC-Segment appears as a signaling data portion of each M/H Group, and FIC-Segments are interleaved on an M/H Subframe basis. (See A/153 Part 2 [31] for further details.)
- The number of FIC-Segments within an M/H Frame is equal to (TNoG x 5).
- An FIC-Segment not carrying a portion of an FIC-Chunk shall be marked as a NULL FIC-Segment in the FIC_segment_type field of its header.
- Every complete FIC-Chunk shall be delivered within a single M/H Frame and shall not extend from one M/H Frame to another.
- The beginning of an FIC-Chunk shall be aligned with the beginning of the payload of the first FIC-Segment carrying the FIC-Chunk.
- Even though FIC-Chunks are sent in M/H Subframes (and interleaved across each), as the total size of a FIC-Chunk can be more than can be sent in one M/H Subframe, FIC-Chunks can overlap M/H Subframe boundaries. Therefore determining what is signaled in each FIC-Chunk as a function of size, the appropriate fraction is determined based on the total M/H Slots available for the delivery of FIC-Chunks in an M/H Frame, not the number of M/H Subframes in which the parts of each FIC-Chunk are sent.
- At least one complete FIC-Chunk that signals the configuration of the next succeeding M/H Frame with valid M/H data shall be delivered at the conclusion of each M/H Frame. The current_next_indicator field of the FIC-Chunk header and of the FIC-Segment headers shall be set to ’0’ (NEXT).
- At least one complete FIC-Chunk that signals the configuration of the current M/H Frame shall be delivered at the beginning of each M/H Frame, if and only if the FIC-Chunk signaling the next succeeding M/H Frame with valid data can be accommodated within the

final 1/2 of the M/H Frame. The current_next_indicator field in the FIC-Chunk header and in the FIC-Segment headers shall be set to ’1’ (CURRENT).

- After placing the FIC-Chunk signaling “NEXT” and the FIC-Chunk signaling “CURRENT”, remaining FIC-Segments left within the M/H Frame boundary may be packed with additional complete FIC-Chunks. If these additional FIC-Chunks are placed, the FIC-Chunks signaling “NEXT” are recommended to be placed as close to the conclusion of the M/H Frame as possible and the FIC-Chunks signaling “CURRENT” are recommended to be placed as close to the beginning of the M/H Frame as possible.
- After placing the complete FIC-Chunks, remaining FIC-Segments left in the M/H Frame may be packed with redundant copies of any segments of any of the complete FIC-Chunks that are placed within the M/H Frame.
- An M/H broadcast shall not contain FIC-Chunks with the same major protocol version and different minor protocol versions. For any given major protocol version, only the current minor protocol version should appear.
- An M/H broadcast may contain FIC-Chunks with different major protocol versions.
- Each distinct FIC-Chunk (taking into account protocol versions and current/next versions) should be delivered as many times as possible. In the case in which there is only a single protocol version, R complete FIC-Chunks should be delivered in each M/H Frame, where the value of R is derived as follows:

$$R = \frac{TNoG \times 5}{\text{Required number of FIC-Segments for the FIC-Chunk}}$$

The bit stream syntax for the FIC-Segment header is provided in Table 6.9.

Table 6.9 Bit Stream Syntax for the FIC-Segment Header

Syntax	No. of Bits	Format
FIC_segment_header() {		
FIC_segment_type	2	uimsbf
reserved	2	’11’
FIC_chunk_major_protocol_version	2	uimsbf
current_next_indicator	1	bslbf
error_indicator	1	bslbf
FIC_segment_num	4	uimsbf
FIC_last_segment_num	4	uimsbf
}		

FIC_segment_type – A two-bit field, which indicates the type of data carried in this FIC-Segment. When the value of this field is set to ‘00’, then it shall indicate that the payload of the FIC-Segment is carrying a portion of an FIC-Chunk. When the value of this field is set to ‘11’, then it shall indicate that the FIC-Segment is a NULL FIC-Segment, not carrying any meaningful data in its payload. Other values are reserved for future extensions.

FIC_chunk_major_protocol_version – A two-bit field, which indicates the major protocol version of the FIC-Chunk which is being carried in part through the payload of this FIC-Segment. The value of this field shall be equal to the value of the FIC_chunk_major_protocol_version field of the FIC_chunk_header().

current_next_indicator – A one-bit field, which indicates the current/next status of the FIC-Chunk which is being carried in part through the payload of this FIC-Segment. The value of this field shall be equal to the value of the current_next_indicator field of the FIC_chunk_header().

error_indicator – This 1-bit indicator shall indicate whether any error was detected in this FIC-Segment. The value '0' shall indicate no error was detected. The value '1' shall indicate that an error was detected.

FIC_segment_num – A 4-bit unsigned integer number field which gives the number of this FIC-Segment. For the first FIC-Segment of an FIC-Chunk, the value of this field shall be set to 0x0. This field shall be incremented by one with each additional FIC-Segment belonging to an FIC-Chunk.

FIC_last_segment_num – A 4-bit unsigned integer number field which gives the number of the last FIC-Segment (i.e., the FIC-Segment with the highest FIC_segment_num) of the complete FIC-Chunk.

7 M/H SERVICE SIGNALING SYSTEM SPECIFICATIONS (NORMATIVE)

This section defines the generic table format and the delivery mechanism for all the M/H Service Signaling tables, the bit stream syntax and semantics for each of these tables, and the bit stream syntax and semantics of the descriptors that may be used within the Service Map Table (SMT-MH) and/or other M/H Service Signaling tables. It also specifies any delivery constraints that apply to the individual tables.

7.1 Table Format

The generic table format common to all M/H Service Signaling tables is given in Table 7.1.

Table 7.1 Generic Table Format used in M/H Service Signaling

Syntax	No. of Bits	Format
MH_service_signaling_table_section() {		
table_id	8	uimsbf
section_syntax_indicator	1	‘0’
private_indicator	1	‘1’
reserved	2	’11’
section_length	12	uimsbf
table_id_extension	16	uimsbf
reserved	2	‘11’
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
MH_service_signaling_data()	*	
}		

table_id – The value of this 8-bit field shall identify the M/H Service Signaling table to which this section belongs.

section_syntax_indicator – This 1-bit field shall be set to ‘0’ to always indicate that this table is derived from the “short” form of the MPEG-2 private section table.

private_indicator – This 1-bit field shall be set to ‘1’.

section_length – A 12-bit field. It specifies the number of remaining bytes this table section immediately following this field. The value in this field shall not exceed 4093 (0xFFD).

table_id_extension – This is a 16-bit field and is table-dependent. It shall be considered to be logically part of the table_id field providing the scope for the remaining fields.

version_number – This 5-bit field is the version number of the entire M/H Service Signaling table. Each table is identified by the combination of table_id and table_id_extension. The version_number shall be incremented by 1 modulo 32 when a change in the information carried within the M/H Service Signaling table occurs.

current_next_indicator – A 1-bit field, which when set to ‘1’ indicates that the MH_service_signaling_section sent is currently applicable. When the current_next_indicator is set to ‘0’, it indicates that the MH_service_signaling_section sent is not yet applicable and shall be the next MH_service_signaling_section with the same section_number, table_id_extension, and table_id to become valid.

section_number – This 8-bit field shall give the section number of this M/H Service Signaling table section. The section_number of the first section in an M/H Service Signaling table shall be 0x00. The section_number shall be incremented by 1 with each additional section in the M/H Service Signaling table.

last_section_number – This 8-bit field shall give the number of the last section (i.e., the section with the highest section_number) of the M/H Service Signaling table of which this section is a part.

7.2 M/H Service Signaling Channel

The M/H Service Signaling Channel for ensemble_protocol_version equal to '00000' shall be a single IP multicast stream carried in each M/H Ensemble. The M/H Service Signaling Channel for ensemble_protocol_version equal to '00000' shall encapsulate data in UDP datagrams and use the IP destination address 224.0.23.60 (IANA has assigned this as AtscSvcSig, see [4]) and destination port 4937/udp (IANA has assigned this to atsc-mh-ssc, see [37]).

For ensemble_protocol_version equal to '00000', an M/H Service Signaling Channel which is unique to a single M/H Broadcast shall use either a unique source address attributed in the global DNS database to the organization which provides the M/H Service Signaling Channel or, in the case of IPv4 datagrams, a source address from the private address range: 10.0.0.0 – 10.255.255.255, 172.16.0.0 – 172.31.255.255, 192.168.0.0 – 192.168.255.255.

An M/H Service Signaling Channel identified with ensemble_protocol_version of '00000' which appears in two or more M/H Broadcasts shall use source addresses attributed in the global DNS database to the organization which provides the Service Signaling Channel.

All the M/H Service Signaling tables, which are listed below, shall only be delivered through the M/H Service Signaling Channel, and no other IP datagrams shall appear in the M/H Service Signaling Channel.

- The Service Map Table (SMT-MH)
- The Guide Access Table (GAT-MH)
- The Cell Information Table (CIT-MH)
- The Service Labeling Table (SLT-MH)
- The Rating Region Table (RRT)

The M/H Service Signaling tables carried through this M/H Service Signaling Channel shall be differentiated by utilizing table_id and table_id_extension included in each table's section header. Figure 7.1 illustrates this mechanism.

7.3 Service Map Table for ATSC-M/H (SMT-MH)

The Service Map Table contains attributes for the M/H Services carried in an M/H Ensemble. Any changes in the contents of the SMT-MH (which would reflect changes in the M/H Service line-up or properties) shall be conveyed in a new SMT-MH carrying an incremented version number. The new SMT-MH shall be inserted into the data stream of the ensemble at the point at which the changes occur. The information contained in the SMT-MH includes service acquisition information for the IP streams that form each M/H Service, such as destination IP address and destination UDP port number. The set of SMT-MH sections in each ensemble shall include information for all M/H Services that are carried wholly or partially in that ensemble.

The SMT-MH shall be carried in ATSC-M/H service signaling sections with table_id 0xDB, and shall obey the syntax and semantics of the Generic Table Format given in Section 7.1.

The following constraints are applied to the delivery of the SMT-MH:

- For each Ensemble, SMT-MH sections describing all the services of that Ensemble shall be included in that Ensemble at least once every RS Frame.
- The SMT-MH sections for each Ensemble shall be differentiated via the ensemble_id included in the section header.

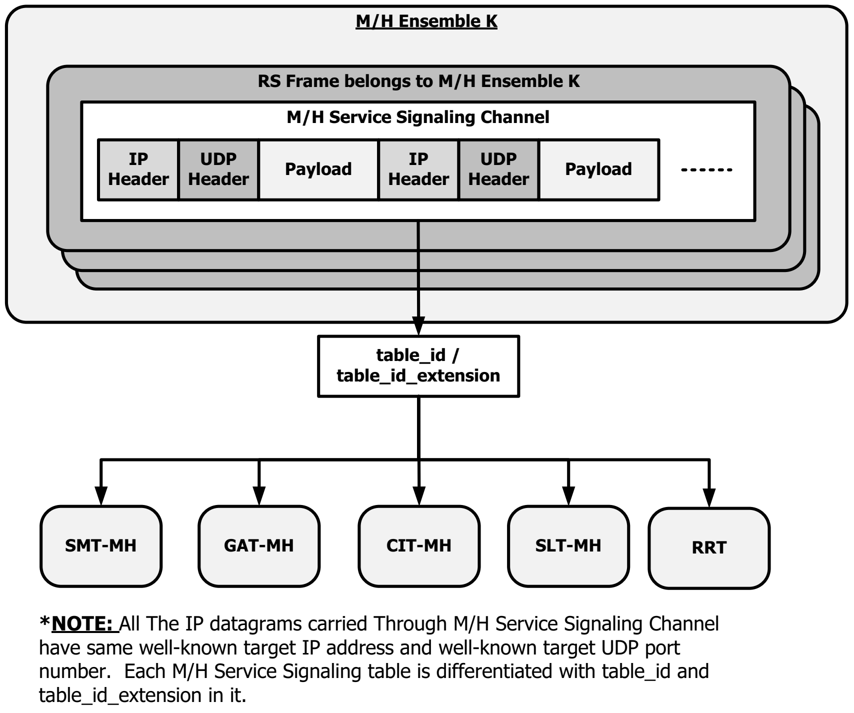
M/H Ensemble K
RS Frame belongs to M/H Ensemble K
M/H Service Signaling Channel
IP Header
UDP Header
Payload
IP Header
UDP Header
Payload
------
table_id / table_id_extension
SMT-MH
GAT-MH
CIT-MH
SLT-MH
RRT
*NOTE: All The IP datagrams carried Through M/H Service Signaling Channel have same well-known target IP address and well-known target UDP port number. Each M/H Service Signaling table is differentiated with table_id and table_id_extension in it.

Figure 7.1 M/H Service Signaling Channel**.**

The bit stream syntax for the Service Map Table sections shall be as shown in Table 7.2.

Table 7.2 Bit Stream Syntax for ATSC-M/H Service Map Table Section

Syntax	No. of Bits	Format
service_map_table_MH_section() {		
table_id	8	0xDB
section_syntax_indicator	1	‘0’
private_indicator	1	‘1’
reserved	2	’11’
section_length	12	uimsbf
table_id_extension {		
SMT_MH_protocol_version	8	uimsbf
ensemble_id	8	uimsbf
}		
reserved	2	‘11’
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
num_MH_services	8	uimsbf

Table 7.2 Bit Stream Syntax for ATSC-M/H Service Map Table Section

for (i=0; i<num_MH_services; i++)		
{		
MH_service_id	16	uimsbf
multi_ensemble_service	2	uimsbf
MH_service_status	2	uimsbf
SP_indicator	1	bslbf
short_MH_service_name_length /* m */	3	uimsbf
short_MH_service_name	16*m	bslbf
reserved	2	'11'
MH_service_category	6	uimsbf
num_components	5	uimsbf
IP_version_flag	1	bslbf
service_source_IP_address_flag	1	bslbf
service_destination_IP_address_flag	1	bslbf
if (service_source_IP_address_flag)		
service_source_IP_address	32 or 128	uimsbf
if (service_destination _IP_address_flag)		
service_destination_IP_address	32 or 128	uimsbf
for (j=0; j<num_components; j++)		
{		
component_source_IP_address_flag	1	bsblf
essential_component_indicator	1	bsblf
component_destination_IP_address_flag	1	bslbf
port_num_count	5	uimsbf
component_destination_UDP_port_num	16	uimsbf
if (component_source_IP_address _flag)		
component_source_IP_address	32 or 128	uimsbf
if (component_destination_IP_address _flag)		
component_destination_IP_address	32 or 128	uimsbf
reserved	4	'1111'
num_component_level_descriptors	4	uimsbf
for (k=0; k<num_component_level_descriptors; k++)		
{		
component_level_descriptor()	var	
}		
}		
reserved	4	'1111'
num_MH_service_level_descriptors	4	uimsbf

Table 7.2 Bit Stream Syntax for ATSC-M/H Service Map Table Section

for (m=0; m<num_MH_service_level_descriptors; m++)		
{		
MH_service_level_descriptor()	var	
}		
}		
reserved	4	‘1111’
num_ensemble_level_descriptors	4	uimsbf
for (n=0; n<num_ensemble_level_descriptors; n++)		
{		
ensemble_level_descriptor()	var	
}		
}		

table_id – An 8-bit unsigned integer number that indicates the type of table section being defined here. For the Service Map Table for ATSC-M/H (SMT-MH), the table_id shall be 0xDB.

SMT_MH_protocol_version – An 8-bit unsigned integer field whose function is to allow, in the future, this Service Map Table to carry parameters that may be structured differently than those defined in the current protocol. At present, the value for the SMT_protocol_version shall be zero. Non-zero values of SMT_protocol_version may be used by a future version of this standard to indicate structurally different tables.

ensemble_id – This 8-bit unsigned integer field shall be the Ensemble ID associated with this M/H Ensemble. The value of this field shall match the associated value in the FIC-Chunk. See Table 6.5.

current_next_indicator – A one-bit indicator, which when set to ‘1’ shall indicate that the Service Map Table sent is currently applicable. When the bit is set to ‘0’, it shall indicate that the table sent is not yet applicable and will be the next table to become valid. This standard imposes no requirement that “next” tables (those with current_next_indicator set to ‘0’) must be sent. An update to the currently applicable table shall be signaled by incrementing the version_number field.

num_MH_services – This 8 bit field shall specify the number of M/H Services in this SMT-MH section.

MH_service_id – A 16-bit unsigned integer number that shall uniquely identify this M/H Service within the scope of this MH Broadcast. The MH_service_id of a service shall not change throughout the life of the service. To avoid confusion, it is recommended that if a service is terminated, then the MH_service_id for the service should not be used for another service until after a suitable interval of time has elapsed. See Annex A for a description of an allocation scheme for MH_service_id values.

multi_ensemble_service – A two-bit enumerated field that shall identify whether the M/H Service is carried across more than one M/H Ensemble. Also, this field shall identify whether or not the

M/H Service can be rendered only with the portion of M/H Service carried through this M/H Ensemble. The value of this field is defined in Table 6.8.

MH_service_status – A 2-bit enumerated field that shall identify the status of this M/H Service. The most significant bit shall indicate whether this M/H Service is active (when set to 1) or inactive (when set to 0) and the least significant bit shall indicate whether this M/H Service is hidden (when set to 1) or not (when set to 0). Hidden services are normally used for proprietary applications, and ordinary receiving devices should ignore them.

SP_indicator – A 1-bit field that shall indicate, when set, that service protection is applied to at least one of the components needed to provide a meaningful presentation of this M/H Service.

short_MH_service_name_length – A three-bit unsigned integer that shall indicate the number of byte pairs in the short_service_name field. This value is shown as 'm' in the No. of Bits column for the short_service_name field. When there is no short name of this M/H service, the value of this field shall be 0.

short_MH_service_name – The short name of the M/H Service, each character of which shall be encoded per UTF-8 [28]. When there is an odd number of bytes in the short name, the second byte of the last of the byte pair per the pair count indicated by the short_service_name_length field shall contain 0x00.

MH_service_category – A 6-bit enumerated type field that shall identify the category of service carried in this M/H Service as defined in Table 7.3. When the value of this field is set to the value which is indicated "Informative only", the value of this field shall be treated as an informative description to the category of service, and the receiver is required to examine the component_level_descriptors() of the SMT-MH to identify the actual category of service carried through this M/H Service. For services that have a video and/or audio component, they shall have an NTP timebase component.

Table 7.3 M/H Service Category

MH_service_category	Meaning
0x00	The service category is not specified by the MH_service_category field. Look in the component_level_descriptor() to identify the category of service.
0x01	**Basic TV (Informative only)** – Look in the component_level_descriptor() to identify the specific category of service.
0x02	**Basic Radio (Informative only)** – Look in the component_level_descriptor() to identify the specific category of service.
0x03	**RI service** – Rights Issuer service as defined in Part #6 [33] of this standard.
0x04	Not specified by the current version of this standard.
0x05	Not specified by the current version of this standard.
0x06	Not specified by the current version of this standard.
0x07	Not specified by the current version of this standard.
0x08	**Service Guide** – Service Guide (Announcement) as defined in Part 4 [44] of this standard.
0x09	Not specified by the current version of this standard.
0x0A	Not specified by the current version of this standard.
0x0B ~ 0xFF	Reserved for future use.

num_components – This 5-bit field specifies the number of IP stream components in this M/H Service.

IP_version_flag – A 1-bit indicator, which when set to '0' shall indicate that service_source_IP_address, service_destination_IP_address, component_destination_IP_address, and component_source_IP_address fields are IPv4 addresses. The value of '1' for this field is reserved for possible future indication that service_source_IP_address, service_destination_IP_address, component_destination_IP_address, and component_source_IP_address fields are for IPv6. Use of IPv6 addressing is not currently defined.

service_source_IP_address_flag – A 1-bit Boolean flag that shall indicate, when set to '1', that an Service source IP address value for this M/H Service is present.

service_destination_IP_address_flag – A 1-bit Boolean flag that indicates, when set to '1', that a service_destination_IP_address value is present, to serve as the default IP address for the components of this M/H Service.

service_source_IP_address – This field shall be present if the service_source_IP_address_flag is set to '1' and shall not be present if the service_source_IP_address_flag is set to '0'. If present, this field shall contain the source IP address of all the IP datagrams carrying the components of this M/H Service, except for the IP datagrams of any components where a component_source_IP_address field is present. The conditional use of the 128 bit-long address version of this field is to facilitate possible use of IPv6 in the future, although use of IPv6 is not currently defined.

service_destination_IP_address – This field shall be present if the service_destination_IP_address_flag is set to '1' and shall not be present if the service_destination_IP_address_flag is set to '0'. If this service_destination_IP_address is not present, then the component_destination_IP_address field shall be present for each component in the num_components loop. The conditional use of the 128 bit-long address version of this field is to facilitate possible use of IPv6 in the future, although use of IPv6 is not currently defined.

component_source_IP_address_flag – A 1-bit Boolean flag that shall indicate, when set to '1', that the component_source_IP_address is present for this component.

essential_component_indicator – A one-bit indicator which, when set to '1', shall indicate that this component is an essential component for the M/H Service. Otherwise, this field indicates that this component is an optional component.

component_destination_IP_address_flag – A 1-bit Boolean flag that shall indicate, when set to '1', that the component_destination_IP_address is present for this component.

port_num_count – This field shall indicate the number of destination UDP ports associated with this UDP/IP stream component. Values of port_num_count greater than one are intended only for components that require multiple UDP/IP streams to carry them. Each stream of such a component shall be assigned a base destination port number. The values of the base destination port numbers shall start from the component_destination_UDP_port_num field and shall assigned sequentially, taking into account that some streams imply the usage of more than one UDP port. For example, an RTP stream would cause an increment by two, to allow for the RTCP stream associated with that RTP stream.

component_destination_UDP_port_num – A 16-bit unsigned integer field, that represents the destination UDP port number for this UDP/IP stream component. For RTP streams, the value of component_destination_UDP_port_num shall be even, and the next higher value shall represent the destination UDP port number of the associated RTCP stream.

component_source_IP_address – This field shall be present if the component_source_IP_address_flag is set to '1' and shall not be present if the component_source_IP_address_flag is set to '0'. When this field is present, the source address of the IP datagrams carrying this component of the MH Service shall match the address in this field. Note: The conditional use of the 128 bit-long address version of this field is to facilitate possible use of IPv6 in the future, although use of IPv6 is not currently defined.

component_destination_IP_address – This field shall be present if the component_destination_IP_address_flag is set to '1' and shall not be present if the component_destination_IP_address_flag is set to '0'. When this field is present, the destination address of the IP datagrams carrying this component of the MH Service shall match the address in this field. When this field is not present, the destination address of the IP datagrams carrying this component shall match the address in the MH_service_destination_IP_address field. The conditional use of the 128 bit-long address version of this field is to facilitate possible use of IPv6 in the future, although use of IPv6 is not currently defined.

num_component_level_descriptors – This 4 bit field specifies the number of component level descriptors for this component.

component_level_descriptor() – One or more descriptors providing additional information for this IP stream component, may be included.

num_MH_service_level_descriptors – This 4 bit field specifies the number of service level descriptors for this service.

MH_service_level_descriptor() – Zero or more descriptors providing additional information for this M/H Service, may be included.

num_ensemble_level_descriptors – This 4 bit field specifies the number of ensemble level descriptors for this ensemble.

ensemble_level_descriptor() – Zero or more descriptors providing additional information for the M/H Ensemble which this SMT-MH describes, may be included.

7.4 Guide Access Table for ATSC-M/H (GAT-MH)

The Guide Access Table for ATSC-M/H (GAT-MH) lists the Service Guide (SG) data sources associated with the M/H Services in the M/H Broadcast, gives information about the SG data service provider for each source, delivery network type of the SG data, and gives access information for each source.

GAT-MH sections describing all SG data services present in the M/H Broadcast shall be included in the M/H Ensemble for which the GAT_ensemble_indicator in the FIC-Chunk is set to ‘1’. It shall be transmitted at least once every minute.

The bit stream syntax for the GAT-MH shall be as shown in Table 7.4.

Table 7.4 Bit Stream Syntax for the Guide Access Table for ATSC-M/H

Syntax	No. of Bits	Format
guide_access_table_MH_section() {		
table_id	8	0xDC
section_syntax_indicator	1	'0'
private_indicator	1	'1'
reserved	2	'11'
section_length	12	uimsbf
table_id_extension {		
GAT_MH_protocol_version	8	uimsbf
reserved	8	'11111111'
}		
reserved	2	'11'
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
num_SG_providers	8	uimsbf
for (i=0; i<num_SG_providers; i++) {		
SG_provider_name_length	8	uimsbf
SG_provider_name_text()	var	bslbf
num_SG_level_descriptors	8	uimsbf
for (j=0; j<num_SG_level_descriptors; j++) {		
SG_level_descriptor()	var	
}		
}		
num_additional_descriptors	8	uimsbf
for (k=0; k<num_additional_descriptors; k++) {		
additional_descriptor()	var	
}		
}		

table_id – An 8-bit unsigned integer number that indicates the type of table section being defined here. For the Guide Access Table for ATSC-M/H (GAT-MH), the table_id shall be 0xDC.

GAT_MH_protocol_version – An 8-bit unsigned integer field whose function is to allow, in the future, this Guide Access Table to carry parameters that may be structured differently than those defined in the current protocol. At present, the value for the GAT_protocol_version shall be zero. Non-zero values of GAT_protocol_version may be used by a future version of this standard to indicate structurally different tables.

num_SG_providers – This 8 bit field shall specify the number of service guide providers described in this GAT-MH section.

SG_provider_name_length – An 8-bit unsigned integer number that shall specify the total length (in bytes) of the SG_provider_name_text() field to follow.

SG_provider_name_text() – The name of the SG provider, in the format of a Unicode character string using UTF-8 encoding [28].

num_SG_level_descriptors – This 8-bit unsigned integer field shall specify the number of Service Guide level descriptors of this GAT-MH.

SG_level_descriptor() – One or more descriptors providing additional information for this Service Guide, may be included.

7.5 Cell Information Table for ATSC-M/H (CIT-MH)

The optional Cell Information Table for ATSC-M/H (CIT-MH), when present, provides carrier frequency information on selected transmitters in adjacent cells that are transmitting services which are the same as, or very similar to, services in the M/H Broadcast where the CIT appears. The purpose of this table is to allow a viewer to continue watching the same service, or a very similar service, when traveling from the coverage area of one M/H transmitter to the coverage area of others. This table only applies to a Multi-Frequency Network environment, and is deprecated for a Single Frequency Network environment.

There are no constraints on the repetition rate of the CIT-MH sections.

The bit stream syntax for the CIT-MH shall be as shown in Table 7.5.

Table 7.5 Bit Stream Syntax for the Cell Information Table for ATSC-M/H

Syntax	No. of Bits	Format
cell_information_table_MH_section() {		
table_id	8	0xDD
section_syntax_indicator	1	‘0’
private_indicator	1	‘1’
reserved	2	’11’
section_length	12	uimsbf
table_id_extension {	16	uimsbf
CIT_MH_protocol_version	8	uimsbf
ensemble_id	8	uimsbf
}		
reserved	2	‘11’
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
num_home_cell_transmitters	8	uimsbf
for (n=0; n<num_home_cell_transmitters; n++) {		

Table 7.5 Bit Stream Syntax for the Cell Information Table for ATSC-M/H

transmitter_latitude	24	signed int
transmitter_longitude	24	signed int
transmitter_AERP	6	uimsbf
transmitter_relative_pattern_depth	2	uimsbf
transmitter_null_positions	8	bslbf
}		
num_MH_services	8	uimsbf
for (i=0; i<num_MH_services; i++) {		
MH_service_id	16	uimsbf
num_cells	8	uimsbf
for (j=0; j<num_cells; j++) {		
cell_latitude	24	signed int
cell_longitude	24	signed int
transport_stream_id	16	uimsbf
transmitter_AERP	6	uimsbf
transmitter_relative_pattern_depth	2	uimsbf
transmitter_null_positions	8	bslbf
PTC_num	8	uimsbf
cell_ensemble_id	8	uimsbf
cell_MH_service_id	16	uimsbf
reserved	4	‘1111’
num_descriptors	4	uimsbf
for (k=0; k<num_descriptors; k++) {		
descriptor()	var	
}		
}		
}		
reserved	4	‘1111’
num_additional_descriptors	4	uimsbf
for (m=0; m<N; m++) {		
additional_descriptor()	var	
}		
}		

table_id – An 8-bit unsigned integer number that indicates the type of table section being defined here. For the Cell Information Table for ATSC-M/H (CIT-MH), the table_id shall be 0xDD.

CIT_MH_protocol_version – An 8-bit unsigned integer field whose function is to allow, in the future, this Cell Information Table to carry parameters that may be structured differently than those defined in the current protocol. At present, the value for CIT_protocol_version shall be zero. Non-zero values of CIT_protocol_version may be used by a future version of this standard to indicate structurally different tables.

ensemble_id – This 8-bit unsigned integer field shall be the Ensemble ID associated with this M/H Ensemble. The value of this field shall match the associated value in the FIC-Chunk. See Table 6.5.

num_home_cell_transmitters – An 8-bit unsigned integer field that shall give the number of transmitters in the home cell whose location is to be announced.

transmitter_latitude – latitude of this home cell transmitter, in ten-thousandths of a degree, with the usual convention on positive and negative latitude values.

transmitter_longitude – longitude of this home cell transmitter, in ten-thousandths of a degree, with the usual convention on positive and negative longitude values.

transmitter_AERP – Effective radiated power in dBk, adjusted for height of antenna center of radiation.

transmitter_relative_pattern_depth – Depth of the largest null in the antenna azimuth pattern in multiples of 8 dB, rounded down to the next lower multiplier value. Any value greater than 24 dB shall be rounded down to 24 dB. A value of '00' may mean no data is available.

transmitter_null_positions – Ordinal direction(s) where the antenna azimuth pattern is 8 dB or more below the peak AERP, as indicated by zeroes in corresponding bit positions. The Northern sector shall be represented by the msb; each succeeding bit shall proceed clockwise around the compass with NE represented by the next most significant bit, and continuing until reaching NW, which shall be represented by the lsb. A value of '1111 1111' may mean no data is available.

num_MH_services – This 8 bit field shall specify the number of M/H Services in this CIT-MH section.

MH_service_id – A 16-bit unsigned integer number that shall identify an M/H Service.

num_cells – This 8 bit field shall specify the number of adjacent transmitters, which transmit an MH service the same as, or very similar to, the service identified by MH_service_id.

transport_stream_id – A 16-bit unsigned integer field in the range of 0x0000 to 0xFFFF that shall identify an M/H Broadcast carrying an M/H service which is to be considered as the same as, or very similar to, the service identified by MH_service_id.

PTC_num – This 8-bit unsigned integer field shall represent the physical transmission channel where the M/H Broadcast represented by the transport_stream_id is transmitted, where the mapping from integer values to frequency bands is given by 47CFR73.603 [6].

cell_ensemble_id – This 8-bit unsigned integer field shall be the Ensemble ID associated with the M/H Ensemble which carries the M/H Service identified by the cell_MH_service_id below.

cell_MH_service_id – A 16-bit unsigned integer number that shall identify the M/H Service in the M/H Broadcast identified by the transport_stream_id above that is to be considered as the same as, or very similar to, the M/H Service identified by the MH_service_id scoping this num_cells loop.

7.6 Service Labeling Table for ATSC-M/H (SLT-MH)

If an M/H receiver finds itself in a new geographical area, with no access to SG data, it could do a very fast frequency scan, looking at only the FIC for each M/H Broadcast it finds, and display to the user a list of available content channels. However, the only information for each content channel in this list would be the MH_service_id.

An M/H receiver in such a situation could also do a much slower frequency scan, looking at the SMT-MH sections in every M/H Ensemble of every M/H Broadcast it finds, and display to the user a more informative list containing the name and service type of each service, as well as the MH_service_id, and perhaps even the title of the current program in each service.

The intent of the optional Service Labeling Table for ATSC-M/H (SLT-MH) is to provide a middle choice, whereby such a receiver could do a relatively fast content channel scan, looking only at the ensemble that carries the SLT-MH in each M/H Broadcast it finds (as signaled in the FIC), and then present a list of the available content channels with the brief name and service type of each content channel, as well as the service ID.

The following constraints apply to the broadcast of the SLT-MH:

- SLT-MH sections, if present, shall be delivered through the M/H Service Signaling Channel of the M/H Ensemble for which the SLT_ensemble_indicator of the FIC-Chunk is set to ‘1’.
- If present, the SLT-MH sections shall be delivered at least once per M/H Frame.

The bit stream syntax for the SLT-MH shall be as shown in Table 7.6.

Table 7.6 Bit Stream Syntax for the Service Labeling Table for ATSC-M/H

Syntax	No. of Bits	Format
service_labeling_table_MH_section() {		
table_id	8	0xDE
section_syntax_indicator	1	‘0’
private_indicator	1	‘1’
reserved	2	’11’
section_length	12	uimsbf
table_id_extension {		
SLT_MH_protocol_version	8	uimsbf
reserved	8	‘11111111’
}		
reserved	2	‘11’
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
num_MH_services	8	uimsbf
for (i=0; i<num_MH_services; i++) {		
reserved	2	‘11’
MH_service_category	6	uimsbf
MH_service_id	16	uimsbf
reserved	5	‘11111’
short_MH_service_name_length /* m */	3	uimsbf
short_MH_service_name	16*m	bslbf
reserved	4	‘1111’
num_descriptors	4	uimsbf
for (m=0; m<num_descriptors; m++) {		
MH_service_descriptor()	var	
}		
}		
}		

table_id – An 8-bit unsigned integer number that indicates the type of table section being defined here. For the Service Labeling Table for ATSC-M/H (SLT-MH), the table_id shall be 0xDE.

SLT_MH_protocol_version – An 8-bit unsigned integer field whose function is to allow, in the future, this Service Labeling Table to carry parameters that may be structured differently than those defined in the current protocol. At present, the value for SLT_protocol_version shall be zero. Non-zero values of SLT_protocol_version may be used by a future version of this standard to indicate structurally different tables.

current_next_indicator – A one-bit indicator, which when set to '1' shall indicate that the Service Labeling Table sent is currently applicable. When the bit is set to '0', it shall indicate that the table sent is not yet applicable and shall be the next table to become valid. This standard imposes no requirement that "next" tables (those with current_next_indicator set to '0') must be sent. An update to the currently applicable table shall be signaled by incrementing the version_number field.

num_MH_services – This 8 bit field shall specify the number of M/H Services in this SLT-MH section.

MH_service_category – A 6-bit enumerated type field that shall identify the type of service carried in this M/H Service as defined in Table 7.3.

MH_service_id – A 16-bit unsigned integer number that shall identify an M/H Service in this M/H Broadcast.

short_MH_service_name_length – A three-bit unsigned integer that shall indicate the number of byte pairs in the short_service_name field. This value is shown as 'm' in the No. of Bits column for the short_service_name field. When there is no short name of this M/H service, the value of this field shall be 0.

short_MH_service_name – The short name of the M/H Service, each character of which shall be encoded per UTF-8 [28]. When there is an odd number of bytes in the short name, the second byte of the last of the byte pair per the pair count indicated by the short_service_name_length field shall contain 0x00. This name shall match the short name for this M/H Service that appears in the SMT-MH.

num_descriptors – A 4-bit unsigned integer number that shall specify the number of descriptors in the following descriptor loop.

MH_service_descriptor() – A descriptor providing information about the MH service.

7.7 Rating Region Table

The Rating Region Table (RRT) specified in ATSC A/65 [3] shall be used when necessary to convey information about the content advisory rating system in use for the M/H Broadcast. All of the specifications and constraints given in A/65 shall apply, except that when the RRT is being transmitted the only constraint on the interval between transmissions is that it shall not be greater than one hour. When the RRT is being transmitted it shall be transmitted in the same ensemble as that designated for the Guide Access Table. It may be transmitted in other ensembles as well.

7.8 M/H-Specific Descriptors

The descriptors described in this section are those currently defined. Others may be defined in the future. Additions to M/H Service Signaling functionality are contemplated in the future and may result in additional descriptors being present in the currently defined M/H Service Signaling tables and/or in possible additional M/H Service Signaling tables. When appropriate, any descriptor constructed in conformance with the tag-length-data structure defined by MPEG-2 [19] may appear in any loop in the tables defined in this standard that contains the word 'descriptor'.

Table 7.7 indicates the descriptor tags for the descriptors defined or referenced in this standard, and their recommended locations in the M/H Service Signaling tables. Exactly one MH_component_descriptor() shall always be present for each component of each service in the SMT-

MH (shown with an "S"). Table 7.7 does not specify the rules governing whether or not a particular descriptor must be present in any given situation. When used, some descriptors shall be in each indicated location (shown with an "M"). Some descriptors also may be present in a second location (shown with an "O"). Asterisks mark the tables where the descriptors may appear without restrictions.

Table 7.7 List and Location of M/H Service Signaling Descriptors

Descriptor Name	**Descriptor Tag**	**M/H Broadcast**			
		SMT-MH			**GAT-MH**
		Component Level	**M/H Service Level**	**Ensemble Level**	**SG Level**
M/H component descriptor	0xBC	S			
M/H rights issuer service descriptor	0xBD		M		
M/H current program descriptor	0xBE		M		
M/H original service Identification descriptor	0xBF		M	O	
Content labeling descriptor	0x36		M		
Caption service descriptor	0x86		M		
Content advisory descriptor	0x87		M		
ATSC private information descriptor	0xAD	*	*	*	
M/H string mapping descriptor	0xB3			M	
Genre descriptor	0xAB		M		
protection descriptor	0xC0		M		
MH SG bootstrap descriptor	0xC1				**S**

7.8.1 M/H Component Descriptor

Exactly one MH_component_descriptor() shall appear in the component descriptor loop of each component of each MH Service in the SMT-MH, and, when the current_next_indicator of the SMT-MH is set to '1', then all parameters in the descriptor shall correspond to the current parameters for that component of the M/H Service[2].

The bit stream syntax for the M/H Component Descriptor shall be as shown in Table 7.8.

2. Note: The critical information commonly conveyed in IP systems via SDP files is instead carried via descriptors.

Table 7.8 Bit Stream Syntax for the M/H Component Descriptor

Syntax	No. of Bits	Format
MH_component_descriptor() {		
descriptor_tag	8	0xBC
descriptor_length	8	uimsbf
component_type	7	uimsbf
component_encryption_flag	1	bsblf
if (component_encryption_flag == ‘1’) {		
num_STKM_streams	8	uimsbf
for (i=0; i<num_STKM_streams; i++) {		
STKM_stream_id	16	uimsbf
}		
}		
transport_parameters_text_length	8	uimsbf
transport_parameters_text()	var	
MH_component_data(component_type)	var	
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xBC, identifying this descriptor as the MH_component_descriptor.

descriptor_length – This 8-bit unsigned integer shall specify the length (in bytes) immediately following this field up to the end of this descriptor.

component_type – This 7-bit field shall identify the encoding format of the component. The value may be any of the values assigned by IANA [42] for the payload_type of an RTP/AVP stream [9], or it may be any of the values in Table 7.9 assigned by ATSC, or it may be a “dynamic value” in the range 96–127. For components consisting of media carried via RTP, the value of this field shall match the value in the payload_type field in the RTP header of the IP stream carrying this component. Note that current assignments are normatively documented in those Parts of this standard which document the particular media and that additional values of the component_type field in the range of 43–71 can be defined in future versions of this standard.

For further information on usage of the “dynamic” values, see Section 7.8.1.8 of this standard.

Table 7.9 Component Type (RTP payload_type)

component_type	Meaning
0-34	Assigned or reserved by IANA, except that 20–24, 27, and 29-30 are unassigned
35	H.264/AVC video stream component or H.264/SVC base layer stream component (assigned by ATSC for M/H use)
36	SVC enhancement layer stream component (assigned by ATSC for M/H use)
37	HE AAC v2 audio stream component (assigned by ATSC for M/H use)
38	FLUTE file delivery session (assigned by ATSC for M/H use)
39	STKM stream component (assigned by ATSC for M/H use)
40	LTKM stream component (assigned by ATSC for M/H use)
41	OMA-RME DIMS stream component (assigned by ATSC for M/H use)
42	NTP timebase stream component (assigned by ATSC for M/H use)
43 – 71	[Unassigned by IANA and reserved by ATSC for possible future M/H use]
72-76	Reserved by IANA
77-95	Unassigned by IANA
96-127	Designated by IANA for dynamic use

Note: IANA states that no further assignments of payload_type values will be made by IANA.

component_encryption_flag – A 1-bit field that indicates that content is encrypted when set to '1'

num_STKM_streams – An 8-bit unsigned integer field that shall identify the number of STKM streams associated with this component.

STKM_stream_id – A 16-bit unsigned integer field that shall identify an STKM stream where keys to decrypt this protected component can be obtained, by reference to the STKM_stream_id in the component descriptor for the STKM stream.

transport_parameters_text_length – This field specifies the length (in bytes) of the decoding_parameters_text() character string. The initial release shall have the value zero.

transport_parameters_text() – A text string that identifies the transport parameters of the component that are to be associated with this MH_component_data(). Future sets of transport constraints (i.e., buffer model. MTU size, etc) that are different from the initial release; when defined, shall be structured and present in this location.

MH_component_data(component_type) – The MH_component_data() element provides the encoding parameters and/or other parameters necessary for rendering this component. The structure of the MH_component_data is determined by the value of component_type field. The MH_component_data() structures corresponding to the currently defined component_type field values are specified in Sections 7.8.1.1 through Section 7.8.1.8. Note that additional MH_component_data() structures can be defined in future versions of this standard.

7.8.2 M/H Component Data for H.264/AVC or H.264/SVC Base Layer (Type 35)

When the component type is 35, the MH_component_data() element shall have the structure shown in Table 7.10[3].

Table 7.10 Bit Stream Syntax for M/H Component Data for H.264/AVC (Type 35)

Syntax	No. of Bits	Format
MH_component_data() {		
profile_idc	8	uimsbf
constraint_set0_flag	1	bslbf
constraint_set1_flag	1	bslbf
constraint_set2_flag	1	bsblf
AVC_compatible_flags	5	bslbf
level_idc	8	uimsbf
AVC_still_present	1	bslbf
AVC_24_hour_picture_flag	1	bslbf
reserved	6	'111111'
}		

profile_idc, **constraint_set0_flag**, **constraint_set1_flag**, **constrant_set2_flag**, **level_idc** – These fields shall be coded according to the semantics for these fields defined in ITU-T Rec. H.264 | ISO/IEC 14496-10 [10].

AVC_compatible_flags – The AVC_compatible_flags shall contain the 5 bits between the constraint_set2_flag and the level_idc field in the Sequence Parameter Set, as defined in ITU-T Rec. H.264 | ISO/IEC 14496-10 [10] and shall have exactly the same semantics of these bits.

AVC_still_present –This is a 1-bit field that shall signal AVC still picture usage. The value '1' shall mean that the associated AVC video stream may include vui_parameters() with fixed_frame_rate_flag ='1'. The value '0' shall mean that the associated AVC video stream only contains vui_parameters() with fixed_frame_rate_flag ='0'.

7.8.3 M/H Component Data for SVC Enhancement Layer (Type 36)

When the component type is 36, the MH_component_data() element shall have the structure shown in Table 7.11[4]. Hereafter in this subsection, the term "base layer" is used to refer to the AVC compatible bitstream subset that contains the lowest temporal/spatial/SNR scalability level.

3. Note: A/153 Part 7 requires these fields to be set to particular values matching those placed in the AVC bitstream. See Part 7, Section 5.
4. Note A/153 Part 7 requires these fields to be set to particular values matching those placed in the AVC bitstream. See Part 7, Section 9.

Table 7.11 Bit Stream Syntax for M/H Component Data for SVC Enhancement Layer (Type 36)

Syntax	No. of Bits	Format
MH_component_data() {		
profile_idc	8	uimsbf
constraint_set0_flag	1	bslbf
constraint_set1_flag	1	bslbf
constraint_set2_flag	1	bsblf
constraint_set3_flag	1	bsblf
SVC_compatible_flags	4	bslbf
level_idc	8	uimsbf
layer_id	8	uimsbf
max_temporal_id	3	uimsbf
max_dependency_id	3	uimsbf
max_quality_id	4	uimsbf
num_directly_dependent_layers	6	uimsbf
for(i = 0; i < num_directly_dependent_layers; i++) {		
directly_dependent_layer_id[i]	8	uimsbf
}		
}		

profile_idc, **constraint_set0_flag**, **constraint_set1_flag**, **constraint_set2_flag**, **constraint_set3_flag** – These five fields shall all be coded according to the semantics for these fields defined in ITU-T Rec. H.264 | ISO/IEC 14496-10 [10].

SVC_compatible_flags – These 4 bits shall be coded exactly as defined for the 4 bits between the constraint_set3_flag and the level_idc field in the Subset Sequence Parameter Set, as defined in ITU-T Rec. H.264 | ISO/IEC 14496-10 [10].

level_idc – This field shall be coded according to the semantics for this field defined in ITU-T Rec. H.264 | ISO/IEC 14496-10 [10].

layer_id – An 8-bit unsigned integer field that specifies the layer identifier of the SVC enhancement layer. The value shall start from ‘1’, where ‘0’ is reserved to indicate the base layer. Furthermore, the value of layer_id for an SVC enhancement layer shall be greater than the layer_id values of all the layers that the SVC enhancement layer depends on.

max_temporal_id – A 3-bit unsigned integer field that specifies the maximum value of temporal_id (a field in the SVC NAL unit header extension) of all VCL NAL units in the SVC enhancement layer. When its value is greater than or equal to 1, the SVC layer includes a temporal (frame rate) enhancement to the base layer, and vice versa. This value indicates the achievable video frame rate level when the enhancement layer and all the layers that it depends on are available for decoding.

max_dependency_id – A 3-bit unsigned integer field that specifies the maximum value of dependency_id (a field in the SVC NAL unit header extension) of all VCL NAL units in the SVC enhancement layer. When its value is greater than or equal to ‘1’, the SVC layer includes a spatial or quality enhancement to the base layer, and vice versa. This value indicates the achievable video resolution level or quality level when the enhancement layer and all the layers it depends on are available for decoding.

max_quality_id – A 4-bit unsigned integer field that specifies the maximum value of quality_id (a field in the SVC NAL unit header extension) of all the VCL NAL units with the maximum value of dependency_id among all VCL NAL units in the SVC enhancement layer. When its value is greater than or equal to '1', the SVC layer includes a medium-grain scalable quality enhancement to the base layer, and vice versa. This value indicates the achievable video quality level when the enhancement layer and all the layers that it depends on are available for decoding.

num_directly_dependent_layers – A 6-bit unsigned integer field that specifies the number of other SVC layers that the current SVC layer directly depends on.

directly_dependent_layer_id – An 8-bit unsigned integer field that specifies the layer identifier of an SVC layer which the current SVC enhancement layer directly depends on.

7.8.4 M/H Component Data for HE AAC v2 Component (Component Type 37)

When the component type is 37, the MH_component_data() element shall have the structure shown in Table 7.12[5].

Table 7.12 Bit Stream Syntax for M/H Component Data for HE AAC v2 (Type 37)

Syntax	No. of Bits	Format
MH_component_data() {		
ISO_639_language_code	3 * 8	
reserved	6	'111111'
RTP_clock_rate	18	uimsbf
constant_duration	16	uimsbf
sampling_rate	4	uimsbf
audio_service_type	4	uimsbf
audio_channel_association	8	bslbf
reserved	4	'1111'
num_configs	4	uimsbf
for (i=0; i<num_configs; i++) {		
profile_level_id	8	uimsbf
num_audio_channels	4	uimsbf
reserved	4	'1111'
config_size	8	uimsbf
if (config_size > 0) {		
config()	config_size * 8	
}		
}		
}		

ISO_639_language_code – This 3-byte (24 bits) field, in conformance with ISO 639.2/B [11], shall specify the language used for this audio stream component. In case of no language specified for this audio stream component, each byte shall have the value 0x00.

RTP_clock_rate – An 18-bit unsigned integer field that shall identify the clock rate of the RTP time stamps, in ticks per second. The clock rate parameter should not be used as an indication for the audio sampling frequency.

constant_duration – A 16-bit unsigned integer field that shall identify the duration of one access unit for this audio stream component, measured in ticks of the clock used for the RTP time stamps.

sampling_rate – A 4-bit unsigned integer field that shall identify the output sampling rate of this audio stream component, in samples per second. The actual output sampling rate can be found in Table 1.16 of ISO/IEC 14496-3 [13], by using the value of the sampling_rate field as the samplingFrequencyIndex.

audio_service_type – A 4-bit unsigned integer that shall identify the audio service type of this HE AAC v2 audio stream component. The meaning of this field is defined in Table 7.13.

Table 7.13 Audio Service Type

audio_service_type	Meaning
0	main audio service: complete main (CM)
1	main audio service: music and effects (ME)
2	associated service: visually impaired (VI)
3	associated service: hearing impaired (HI)
4	associated service: dialogue (D)
5	associated service: commentary (C)
6	[Reserved for future ATSC use]
7	associated service: voice over (VO)
8	main audio service: karaoke
9 – 15	[Reserved for future ATSC use]

audio_channel_association – If the audio_service_type is a main audio service (types 0, 1, or 8), then the high order five bits of this field shall be reserved, and the low order three bits of this field shall contain a number in the range 0–7, which shall identify this audio service. Different main services shall be identified with different numbers. This value is used as an identifier to link associated services with particular main services. If the audio_service_type is an associated service type (types 2–7), then each bit (0–7) of this field identifies the main service(s) with which this associated service can be associated. For example, the left-most bit, bit 7, indicates whether this associated service may be reproduced along with main service number 7. If the bit has a value of 1, the service is associated with main service number 7. If the bit has a value of 0, the service is not associated with main service number 7.

5. Note: A/153 Part 8 requires these fields to be set to particular values matching those placed in the HE AAC bitstream. See Part 8, Section 5.

num_configs – A 4-bit unsigned integer that shall identify how many decoder configuration records are present in this descriptor. The value of this field shall not be set to '0'. Decoders only capable of decoding HE AAC v2 shall use the first configuration record to initialize.

profile_level_id – An 8-bit unsigned integer field that shall identify a decimal representation of the MPEG-4 Profile Level indication. This parameter shall be used to indicate the MPEG-4 Profile and Level combination that is needed to render the audio stream with which that this descriptor is associated.The MPEG-4 Audio Profiles and Levels are listed in Table 1.12 of ISO/IEC 14496-3 with Corrigendum No. 2 [13].

num_audio_channels – A 4-bit unsigned integer field that shall identify the number of audio channels that will be used by a decoder, according to the value of the profile_level_id parameter, to render the audio stream component.

config_size – An 8-bit unsigned integer field that shall identify the size in bytes of the following decoder configuration octet string.

config() – An octet string that shall express the media payload configuration. For MPEG-4 Audio streams, config() represents the audio object type specific decoder configuration data AudioSpecificConfig() as defined in ISO/IEC 14496-3 [13].

7.8.5 M/H Component Data for FLUTE File Delivery (Type 38)

The MH_component_data() for component type 38 is modeled on the description of a FLUTE session given in the IETF Internet-draft-mehta-rmt-flute-sdp-05.txt ("SDP Descriptors for FLUTE" – January, 27, 2006) [32], but it does not normatively reference that description.

The source IP address of the FLUTE session is not explicitly represented in this component data. However, it shall be provided at the service or component level whenever a FLUTE component is specified.

When the component type is 38, the MH_component_data() element shall have the structure shown in table 7.14.

Table 7.14 Bit Stream Syntax for M/H Component Data for FLUTE File Delivery (Type 38)

Syntax	No. of Bits	Format
MH_component_data() {		
TSI	16	uimsbf
session_start_time	32	uimsbf
session_end_time	32	uimsbf
reserved	5	'11111'
tias_bandwidth_indicator	1	bslbf
as_bandwidth_indicator	1	bslbf
FEC_OTI_indicator	1	bslbf
if (tias_bandwidth_indicator == '1') {		
tias_bandwidth	16	uimsbf
}		
if (as_bandwidth_indicator == '1') {		
as_bandwidth	16	uimsbf
}		
if (FEC_OTI_indicator == '1') {		
FEC_encoding_id	8	uimsbf
FEC_instance_id	16	uimsbf
}		
}		

TSI – A 16-bit unsigned integer field, which shall be the Transport Session Identifier (TSI) of the FLUTE session.

session_start_time – A 32-bit unsigned integer quantity that shall be the seconds part of the NTP timestamp corresponding to the time at which the FLUTE session starts. This timestamp is computed according to the specifications in NTPv4 [21]. The timescale shall be the same as that signaled in the time of day as defined in Section 8. If the value of this field is set to all-zero, then it shall be interpreted to mean that the session has already started.

session_end_time – A 32-bit unsigned integer quantity that shall be the seconds part of the NTP timestamp corresponding to the time at which the FLUTE session ends. This timestamp is computed according to the specifications in NTPv4 [21]. The timescale shall be the same as that signaled in the time of day as defined in Section 8. If the value of this field is set to all-zero, then it shall be interpreted to mean that the session continues indefinitely.

tias_bandwidth_indicator – A 1-bit field that flags the inclusion of TIAS bandwidth information. This bit shall be set to '1' to indicate the TIAS bandwidth field is present, and it shall be set to '0' to indicate the TIAS bandwidth field is absent.

as_bandwidth_indicator – A 1-bit field that flags the inclusion of AS bandwidth information. This bit shall be set to '1' to indicate the AS bandwidth field is present, and it shall be set to '0' to indicate the AS bandwidth field is absent.

FEC_OTI_indicator – A 1-bit indicator that indicates whether FEC Object Transmission Information is provided.

tias_bandwidth – This value shall be one one-thousandth of the Transport Independent Application Specific maximum bandwidth as defined in RFC 3890 [14], rounded up to the next highest integer if necessary.

as_bandwidth – This value shall be the Application Specific maximum bandwidth as defined in RFC 4566 [15]. One kilobit per second shall be interpreted as 1000 bits per second.

FEC_encoding_id – FEC encoding ID used in this FLUTE session, as defined in RFC 3926 [16].

FEC_instance_id – FEC instance ID used in this FLUTE session, as defined in RFC 3926 [16].

7.8.6 M/H Component Data for STKM (Type 39)

When the component type is 39, the MH_component_data() element shall have the structure shown in Table 7.15.

Table 7.15 Bit Stream Syntax for M/H Component Data for STKM (Type 39)

Syntax	No. of Bits	Format
MH_component_data() {		
MH_service_protection_version	4	uimsbf
encryption_type	8	uimsbf
kms_type	3	uimsbf
reserved	1	’1’
STKM_stream_id	16	uimsbf
if (kms_type == 0x0) {		
reserved	5	‘11111’
base_CID_indicator	1	bslbf
srv_CID_extension_indicator	1	bslbf
prg_CID_extension_indicator	1	bsblf
if (base_CID_indicator == ‘1’) {		
base_CID_text_length	8	uimsbf
base_CID_text()	var	
}		
if (srv_CID_extension_indicator == ‘1’) {		
srv_CID_extension	8	uimsbf
}		
if (prg_CID_extension_indicator == ‘1’) {		
prg_CID_extension	8	uimsbf
}		
num_rights_issuers	8	uimsbf
for (i=0; i<num_rights_issuers) {		
URI_id_flag	1	bslbf
If (URL_id_flag == ‘1’)		
rights_issuer_URI_id	7	uimsbf
else {		
rights_issuer_URI_length	7	uimsbf
rights_issuer_URI()	var	
}		
}		
}		
else {		
}		
}		

MH_service_protection_version – A 4-bit field that identifies the version of the ATSC-M/H service protection.

encryption_type – An 8-bit field that identifies the encryption algorithm used to protect the content for which this STKM stream controls access. This field shall be interpreted according to the OMA BCAST Service Guide specification [9] for the EncryptionType element of the Access fragment.

kms_type – A 4-bit unsigned integer that identifies the type of Key Management System(s) (KMS). This field shall be interpreted according to the OMA BCAST Service Guide specification [9] for the kmsType attribute of the Access fragment. The value of this field shall be set to "0x0" to conform to this version of standard.

STKM_stream_id – A 16-bit unsigned integer field that shall uniquely identify this particular key stream within an M/H Service scope. This identifier is referenced by the component descriptors of protected components to indicate where keys can be found to decrypt the components.

base_CID_indicator – A 1-bit indicator, when set to '1', indicates the base_CID_text() is included.

srv_CID_extension_indicator – A 1-bit indicator, when set to '1', indicates the srv_CID_extension is included.

prg_CID_extension_indicator – A 1-bit indicator, when set to '1', indicates the prg_CID_extension is included.

base_CID_text_length – This field specifies the length (in bytes) of the base_CID_text() character string.

base_CID_text() – The BaseCID, which is part of the Service or Program CID used to identify the corresponding asset within an OMA DRM 2.0 Rights Object. Upon reception of a STKM, the receiver can assemble the service_CID/program_CID/BCI and look up the SEK or PEK (wrapped inside a LTKM) as described in Section 5.5.1 of OMA-TS-BCAST_SrvCntProtection [17].

srv_CID_extension – This 8-bit unsigned integer field specifies the most significant byte of the service_CID_extension, which is part of the Service CID used to identify the corresponding asset within an OMA DRM 2.0 Rights Object. Upon reception of a STKM, the receiver can assemble the service_CID and look up the SEK (wrapped inside a LTKM) as described in Section 5.5.1 of OMA-TS-BCAST_SrvCntProtection [17]. This parameter provides the value of the most significant byte of the service_CID_extension in the corresponding STKM stream. This field shall be present if the protected component has multiple DRM profile STKM streams for a rights_issuer_URI() and with different values of service_CID_extension. Within each rights_issuer_URI(), for each protected component, the value of srv_CID_extension shall be unique for each STKM stream.

prg_CID_extension – This 8-bit unsigned integer field specifies the most significant byte of the program_CID_extension, which is part of the Service CID used to identify the corresponding asset within an OMA DRM 2.0 Rights Object. Upon reception of a STKM, the receiver can assemble the program_CID/BCI and look up the PEK (wrapped inside a LTKM) as described in Section 5.5.1 of OMA-TS-BCAST_SrvCntProtection [17]. This parameter provides the value of the most significant byte of the program_CID_extension in the corresponding STKM stream. This field shall be present if the protected component has multiple DRM profile STKM streams for a rights_issuer_URI() and with the different value of program_CID_extension. Within each rights_issuer_URI(), for each protected component, the value of prg_CID_extension shall be unique within for each STKM stream.

num_rights_issuers – The value of this field shall be a positive integer giving the number of Rights Issuers using this STKM stream to deliver decryption keys.

URI_id_flag – This 1-bit field shall be set to '1' when it is followed by a reference to the Rights Issuer URI in an MH String Mapping Descriptor, and it shall be set to '0' when it is followed by the Rights Issuer URI in the form of a string.

rights_issuer_URI_id – This 7-bit integer shall match the value of a string_id field in the MH_string_mapping_descriptor() of this ensemble, thereby identifying the corresponding string_value() field in that descriptor as the identifier of the service provider in the form of a URI.

rights_issuer_URI_length – This 7-bit field shall specify the length (in bytes) of the rights_issuers_URI() character string.

rights_issuer_URI() – This character string shall be the identifier of the service provider in the form of a URI.

7.8.7 M/H Component Data for LTKM (Type 40)

When the component type is 40, the MH_component_data() element shall have the structure shown in Table 7.16.

Table 7.16 Bit Stream Syntax for M/H Component Data for LTKM (Type 40)

Syntax	**No. of Bits**	**Format**
MH_component_data(){		
MH_service_protection_version	4	uimsbf
kms_type	3	uimsbf
LTKM_stream_type	1	bslbf
if (LTKM_stream_type == '0') {		
URI_id_flag	1	bslbf
if (URI_id_flag == '1')		
rights_issuer_URI_id	7	uimsbf
else {		
rights_issuer_URI_length	7	uimsbf
rights_issuer_URI()	var	
}		
}		
}		

MH_service_protection_version – A 4-bit field that identifies the version of the ATSC-M/H service protection.

kms_type – A 3-bit unsigned integer that identifies the type of Key Management System(s) (KMS). The value of this field shall be set to "0x00" to conform this version of standard.

LTKM_stream_type – A one bit indicator, when set to '1', indicates that this LTKM stream component is an ad-hoc RI stream component of a right issuer service.

URI_id_flag – This 1-bit field shall be set to '1' when it is followed by a reference to the Rights Issuer URI in an MH String Mapping Descriptor, and it shall be set to '0' when it is followed by the Rights Issuer URI in the form of a string.

rights_issuer_URI_id – This 7-bit integer shall match the value of a string_id field in the MH_string_mapping_descriptor() of this ensemble, thereby identifying the corresponding string_value() field in that descriptor as the identifier of the service provider in the form of a URI.

rights_issuer_URI_length – This 7-bit field shall specify the length (in bytes) of the rights_issuer_URI() character string.

rights_issuer_URI() – This character string shall be the identifier of the service provider in the form of a URI.

7.8.8 M/H Component Data for OMA-RME DIMS (Type 41)

When the component type is 41, the MH_component_data() element shall have the structure shown in table 7.17.

Table 7.17 Bit Stream Syntax for the M/H Component Data for OMA-RME DIMS (Type 41)

Syntax	No. of Bits	Format
MH_component_data(){		
version_profile	8	uimsbf
level	8	uimsbf
}		

version_profile – An 8-bit unsigned integer field that shall represent the version and the profile of the DIMS stream.

level – An 8-bit unsigned integer field that shall represent the level of the DIMS stream.

7.8.9 M/H Component Data for NTP timebase (Type 42)

When the component type is 42, the MH_component_data() element shall have the structure shown in table 7.18.

Table 7.18 Bit Stream Syntax for the M/H Component Data for NTP timebase (Type 42)

Syntax	No. of Bits	Format
MH_component_data(){		
version	8	uimsbf
}		

version – An 8-bit unsigned integer field that shall represent the version of the NTP packet stream. The value zero shall mean NTPv4 as defined in this document. All other values are ATSC reserved.

7.8.10 M/H Component Data for Dynamic Range Type (Type 96 – 127)

The dynamic range 96–127 of component_type values are intended for media formats that become widely used in IP environments, but are not among the formats for which IANA has assigned specific payload_type values and are not among the formats for which ATSC has assigned specific component_type values in Table 7.9 of this standard.

When the component type is in the range 96–127, the MH_component_data() element shall have the structure shown in Table 7.19. This structure allows a textual description of the MIME type and the encoding parameters for such a component type, in a form similar to what would be used in a Session Description Protocol (SDP) file [15].

Table 7.19 Bit Stream Syntax for M/H Component Data for Dynamic Range Type (Type 96 – 127)

Syntax	No. of Bits	Format
MH_component_data() {		
reserved	4	‘1111’
general_media_type	4	uimsbf
if (general_media_type == 1) {		
ISO_639_language_code	3*8	
}		
media_type_text_length	8	uimsbf
media_type_text()	var	
decoding_parameters_text_length	8	uimsbf
decoding_parameters_text()	var	
}		

general_media_type – A 4-bit unsigned integer field that identifies the general media type of the component. The values represent media types as specified in Section 8.2.1 of the IETF Session Description Protocol (SDP) [15]. The specific value assignment of this field is provided in Table 7.20.

Table 7.20 General Media Type

general_media_type	SDP Media Type	Meaning
0x0	video	Video stream
0x1	audio	Audio stream
0x2	text	Text stream
0x3	application	Application data stream
0x4	message	Message stream
0x5-0xF		Reserved for future ATSC use

ISO_639_language_code – This 3-byte (24 bits) field, in conformance with ISO 639.2/B [11], shall specify the language used for this audio stream component. In case of no language specified for this audio stream component, each byte shall have the value 0x00.

media_type_text_length – This field shall specify the length (in bytes) of the media_type_text() character string.

media_type_text() – A text string that identifies the media sub-type and the encoding clock rate of the component that this MH_component_type_data() is associated with, and for audio streams optionally the number of audio channels. This text string shall conform to the syntax of the "<encoding name>/<clock rate> [/<encoding parameters>]" portion of an "a=rtpmap" entry in an SDP message, as specified in the IETF SDP RFC [15].

decoding_parameters_text_length – This field specifies the length (in bytes) of the decoding_parameters_text() character string.

decoding_parameters_text() – A text string that identifies the encoding parameters of the component that this MH_component_type_data() is associated with. This text string shall conform to the syntax of the "<format specific parameters>" portion of an "a=fmtp" entry in an SDP message, as specified in the IETF SDP RFC [15].

7.8.11 M/H Rights Issuer Service Descriptor

The MH_rights_issuer_service_descriptor() shall be used as an MH_service_level_descriptor() of the SMT-MH for a rights issuer service. This descriptor provides the necessary information to support efficient rights issuer service carriage. The bit stream syntax is given in Table 7.21. For further details about the rights issuer service, see ATSC A/153 Part 6 [33].

Table 7.21 Bit Stream Syntax for the M/H Rights Issuer Service Descriptor

Syntax	No. of Bits	Format
MH_rights_issuer_service_descriptor() {		
descriptor_tag	8	0xBD
descriptor_length	8	uimsbf
MH_service_protection_version	4	uimsbf
kms_type	3	uimsbf
reserved	1	'1'
URI_id_flag	1	bslbf
If (URL_id_flag == '1')		
rights_issuer_URI_id	7	uimsbf
else {		
rights_issuer_URI_length	7	uimsbf
rights_issuer_URI()	var	
}		
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xBD, identifying this descriptor as MH_rights_issuer_service_descriptor.

descriptor_length – This 8-bit unsigned integer shall specify the length (in bytes) immediately following this field up to the end of this descriptor.

MH_service_protection_version – The initial value is 0x00.

kms_type – A 3-bit unsigned integer that shall identify the type of Key Management System(s) (KMS). The value of this field shall be set to "0x00" to conform to this version of this standard.

URI_id_flag – This 1-bit field shall be set to '1' when it is followed by a reference to the Rights Issuer URI in an MH String Mapping Descriptor, and it shall be set to '0' when it is followed by the Rights Issuer URI in the form of a string.

rights_issuer_URI_id – This 7-bit integer shall match the value of a string_id field in the MH_string_mapping_descriptor() of this ensemble, thereby identifying the corresponding string_value() field in that descriptor as the identifier of the service provider in the form of a URI.

rights_issuer_URI_length – This field shall specify the length (in bytes) of the rights_issuer_URI() character string.

rights_issuer_URI() – This field shall contain the identifier of the service provider in the form of a URI.

7.8.12 M/H Current Program Descriptor

The MH_current_program_descriptor(), when present, shall be used to provide basic information about the current program in this M/H Service (current program start time, duration, and title).

The bit stream syntax for the M/H Current Program Descriptor shall be as shown in Table 7.22.

Table 7.22 Bit Stream Syntax for the M/H Current Program Descriptor

Syntax	No. of Bits	Format
MH_current_program_descriptor() {		
descriptor_tag	8	0xBE
descriptor_length	8	uimsbf
current_program_start_time	4*8	uimsbf
current_program_duration	3*8	uimsbf
title_length	8	uimsbf
title_text()	var	
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xBE, identifying this descriptor as MH_current_program_descriptor.

descriptor_length – This 8-bit unsigned integer shall specify the length (in bytes) immediately following this field up to the end of this descriptor.

current_program_start_time – A 32-bit unsigned integer quantity that shall be the seconds part of the NTP timestamp corresponding to that shall represent the start time of the current program. This timestamp is computed according to the specifications in NTPv4 [21]. The timescale shall be the same as that signaled in the time of day as defined in Section 8.

current_program_duration – This field shall give the duration of this program in seconds.

title_length – This field shall specify the length (in bytes) of the title_text(). Value 0 means that no title is given for this program.

title_text() – This field shall give the program title in the format of a Multiple String Structure as defined in A/65 [3].

7.8.13 M/H Original Service Identification Descriptor

The optional MH_original_service_id_descriptor(), when present, shall be used as an MH_service_level_descriptor() in the SMT-MH. This descriptor provides the original MH_service_id of the M/H Service, when its MH_service_id has been changed for some reason, for example to rebrand the service for local broadcast.

The bit stream syntax for the M/H Original Service Identification Descriptor shall be as shown in Table 7.23.

Table 7.23 Bit Stream Syntax for the M/H Original Service Identification Descriptor

Syntax	No. of Bits	Format
MH_original_service_id_descriptor() {		
descriptor_tag	8	0xBF
descriptor_length	8	uimsbf
MH_original_service_id	16	uimsbf
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xBF, identifying this descriptor as MH_original_service_id_descriptor.

descriptor_length – This 8-bit unsigned integer specifies the length (in bytes) immediately following this field up to the end of this descriptor.

MH_original_service_id – A 16-bit unsigned integer number that represents the MH_service_id of the M/H Service as it was assigned by the originating delivery system.

7.8.14 Content Labeling Descriptor

The content_labeling_descriptor() (descriptor_tag 0x36) of ISO/IEC 13818-1 [18] shall be used as a MH_service_level_descriptor() in the SMT-MH to identify and label the content currently being broadcast in the Service. The structure and semantics of the descriptor shall conform to the ATSC Standard for Content Identification and Labeling [19].

7.8.15 Caption Service Descriptor

The caption_service_descriptor() (descriptor_tag 0x86) of ATSC A/65 [3], when present, shall be used as a MH_service_level_descriptor() in the SMT-MH. The caption service descriptor provides closed captioning information, such as closed captioning type and language code for the current content, if one or more closed captioning services are provided. If there is no captioning currently present in the content for an M/H Service, the caption service descriptor shall not be present in the SMT-MH for that service.

7.8.16 Content Advisory Descriptor

The content_advisory_descriptor() (descriptor_tag 0x87) defined in ATSC A/65 [3] may appear as an MH_service_level_descriptor() in the SMT-MH. When this descriptor is used for rating region 1 or rating region 2, the values that appear in the descriptor and their semantics shall conform to the specifications in CEA-766 [20].

7.8.17 ATSC Private Information Descriptor

The ATSC_private_information_descriptor() (descriptor_tag 0xAD), of ATSC A/53 Part 3 [8] may be in any descriptor loop.

7.8.18 ATSC Genre Descriptor

ATSC genre descriptors, as defined in A/65 [3], may appear in the MH Service level descriptor loop in the SMT and/or in the SLT.

7.8.19 M/H String Mapping Descriptor

An MH_string_mapping-descriptor() may be used to provide a mapping from one or more integers to one or more corresponding strings. Multiple instances of the MH_string_mapping_descriptor() may appear in the same descriptor loop. It is anticipated that one important use will be in the descriptor loop for some ensembles.

The bit stream syntax for the M/H String Mapping Descriptor shall be as shown in Table 7.24.

Table 7.24 Bit Stream Syntax for the M/H String Mapping Descriptor

Syntax	No. of Bits	Format
MH_string_mapping_descriptor() {		
descriptor_tag	8	0xB3
descriptor_length	8	uimsbf
num_entries	8	uimsbf
for (j=0;j<num_entries; J++) {		
string_id	8	uimsbf
string_length	8	uimsbf
string_value()	var	
}		
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xBC, identifying this descriptor as the MH_string_mapping_descriptor.

descriptor_length – This 8-bit unsigned integer shall specify the length (in bytes) immediately following this field up to the end of this descriptor.

num_entries – This 8-bit unsigned integer shall specify the number of id/value pairs in this descriptor.

string_id – This 8-bit unsigned integer shall identify the text in the parameter string_value. Each string_id value shall be unique within the scope of all the String Mapping Descriptors appearing in the SMT of a single Ensemble.

string_length – This 8-bit unsigned integer shall specify the length in bytes of the string_value() field.

string_value() – A string of bytes.

7.8.20 Protection Descriptor

A protection_descriptor() shall appear in the service descriptor loop for each service that has service protection applied. This descriptor indicates the option used for filtering encrypted and clear packets of a multicast component stream of a given destination IP address.

The bit stream syntax for the Protection Descriptor shall be as shown in Table 7.25.

Table 7.25 Bit Stream Syntax for the Protection Descriptor

Syntax	No. of Bits	Format
protection_descriptor() {		
descriptor_tag	8	0xC0
descriptor_length	8	uimsbf
MH_service_protection_filtering_option	1	uimsbf
reserved	7	'1111111'
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xC0, identifying this descriptor as the protection_descriptor.

descriptor_length – This 8-bit unsigned integer shall specify the length (in bytes) immediately following this field up to the end of this descriptor.

MH_service_protection_filtering_option – A 1-bit indicator, which shall indicate the use of option A or B, as defined in Section 6.2 of A/153 Part 6 [33]. '0' shall indicate the use of the "Option A", '1' indicates the use of the "Option B". Access to ATSC services using Option A will not be possible for decoders implementing only Option B so such decoders should not attempt to decrypt such services.

Note: This descriptor is expected to be modified to signal additional content or service protection characteristics when those are defined. These may include signaling of Digital Rights Management assertions.

7.8.21 MH_SG_bootstrap_descriptor

An MH_SG_bootstrap_descriptor() shall appear in the Service Guide level descriptor loop of each Service Guide in the GAT-MH and provide the parameters that are necessary for bootstrapping a Service Guide.

The bit stream syntax for the M/H SG Bootstrap Descriptor shall be as shown in Table 7.26.

Table 7.26 Bit Stream Syntax for the M/H SG Bootstrap Descriptor

Syntax	No. of Bits	Format
MH_SG_bootstrap_descriptor() {		
descriptor_tag	8	0xC1
descriptor_length	8	uimsbf
reserved	3	'111'
SG_delivery_network_type	5	uimsbf
SG_bootstrap_data(SG_delivery_network_type)	var	
}		

descriptor_tag – This 8-bit unsigned integer shall have the value 0xC1, identifying this descriptor as the MH_SG_bootstrap_descriptor.

descriptor_length – This 8-bit unsigned integer shall specify the length (in bytes) immediately following this field up to the end of this descriptor.

SG_delivery_network_type – A 5-bit unsigned integer field that identifies the type of the delivery network where the SG data is delivered, as defined in Table 7.27.

Table 7.27 SG Delivery Network Type

SG_delivery_network_type	Meaning
0x00	Some SG fragments are delivered through the same M/H Broadcast where this GAT-MH is delivered.
0x01	Some SG fragments are delivered through a different M/H Broadcast from the M/H Broadcast where this GAT-MH is delivered.
0x02	Some SG fragments are delivered through a non-M/H IP-based broadcast channel.
0x03	Some SG fragments are delivered through an IP-based interaction channel.
0x04 – 0x1F	Reserved for future use.

Note: The terms "broadcast channel" and the "interaction channel" have the same meanings as used in the OMA-BCAST standard.

SG_bootstrap_data(SG_delivery_network_type) – The SG_bootstrap_data() element provides the parameters necessary for bootstrapping a Service Guide. The structure of the SG_bootstrap_data() is determined by the value of SG_delivery_network_type field. The SG_bootstrap_data() structures corresponding to the currently defined SG_delivery_network_type field values are specified in Table 7.28 through Table 7.31. Note that additional SG_bootstrap_data() structures can be defined in future versions of this standard.

7.8.21.1 M/H SG Bootstrap Data for an SG on the same M/H Broadcast

When the SG_delivery_network_type field is set to 0x00, identifying the MH_SG_bootstrap_descriptor() is signaling the Service Guide delivered through the same M/H Broadcast where the GAT-MH is delivered, the SG_bootstrap_data() element shall have the structure shown in Table 7.28.

Table 7.28 Bit Stream Syntax for SG Bootstrap Data for the Same M/H Broadcast (Type 0x00)

Syntax	No. of Bits	Format
SG_bootstrap_data() {		
MH_service_id	16	uimsbf
announcement_channel_tsi	16	uimsbf
}		

MH_service_id – A 16-bit unsigned integer number that shall be the Service ID of an SG data service for this service provider.

announcement_channel_tsi – A 16-bit unsigned integer number that shall be the TSI of the FLUTE session which is the Announcement Channel of the SG data service.

7.8.21.2 M/H SG Bootstrap Data for an SG on a different M/H Broadcast

When the SG_delivery_network_type field is set to 0x01, identifying the MH_SG_bootstrap_descriptor() is signaling the Service Guide delivered through a different M/H Broadcast from the M/H Broadcast where the GAT-MH is delivered, the SG_bootstrap_data() element shall have the structure shown in Table 7.29.

Table 7.29 Bit Stream Syntax for SG Bootstrap Data for the Different M/H Broadcast (Type 0x01)

Syntax	No. of Bits	Format
SG_bootstrap_data() {		
transport_stream_id	16	uimsbf
MH_service_id	16	uimsbf
announcement_channel_tsi	16	uimsbf
}		

transport_stream_id – This 16 bit field shall contain the TSID of the broadcaster in which the FLUTE session which is the Announcement Channel of the SG data service identified by the announcement_channel_tsi is sent.

MH_service_id – A 16-bit unsigned integer number that shall be the Service ID of an SG data service for this service provider.

announcement_channel_tsi – A 16-bit unsigned integer number that shall be the TSI of the FLUTE session which is the Announcement Channel of the SG data service.

7.8.21.3 M/H SG Bootstrap Data for an SG on a non-M/H IP-based Broadcast Channel

When the SG_delivery_network_type field is set to 0x02, identifying the MH_SG_bootstrap_descriptor() is signaling the Service Guide delivered through a non-M/H IP-based broadcast channel, the SG_bootstrap_data() element shall have the structure shown in Table 7.30.

Table 7.30 Bit Stream Syntax for SG Bootstrap Data for the non-M/H IP-based Broadcast Channel (Type 0x02)

Syntax	No. of Bits	Format
SG_bootstrap_data() {		
reserved	6	'111111'
IP_version_flag	1	bslbf
source_IP_address_flag	1	bslbf
if (source_IP_address_flag)		
announcement_channel_source_IP_address	32 or 128	uimsbf
announcement_channel_destination_IP_address	32 or 128	uimsbf
announcement_channel_destination_UDP_port_num	16	uimsbf
announcement_channel_TSI	16	uimsbf
}		

IP_version_flag – A 1-bit indicator, which when set to '0' shall indicate that the announcement_channel_source_IP_address, announcement_channel_destination_IP_address fields are IPv4 addresses and when set to '1' shall indicate announcement_channel_source_IP_address, announcement_channel_destination_IP_address fields are IPv6 addresses.

source_IP_address_flag – A 1-bit Boolean flag that shall indicate, when set, that a source IP address value for the SG announcement channel is present to indicate a source specific multicast.

announcement_channel_source_IP_address – This field shall be present if the source_IP_address_flag is set to '1' and shall not be present if the source_IP_address_flag is set to '0'. If present, this field shall contain the source IP address of all the IP datagrams carrying the SG announcement channel. The conditional use of 128 bit-long address version of this field is to facilitate use of IPv6.

announcement_channel_destination_IP_address – This field shall contain the IP multicast destination address of the SG announcement channel.

announcement_channel_destination_UDP_port_num – A 16-bit unsigned integer field that represents the destination UDP port number for the SG announcement channel.

announcement_channel_tsi – A 16-bit unsigned integer number that shall be the TSI of the FLUTE session which is the Announcement Channel of the SG data service.

7.8.21.4 M/H SG Bootstrap Data for an SG on an IP-based Interaction Channel

When the SG_delivery_network_type field is set to 0x03, identifying the MH_SG_bootstrap_descriptor() is signaling the Service Guide delivered through an IP-based interaction channel, the SG_bootstrap_data() element shall have the structure shown in Table 7.31.

Table 7.31 Bit Stream Syntax for SG Bootstrap Data for the Interaction Channel (Type 0x03)

Syntax	No. of Bits	Format
SG_bootstrap_data() {		
SG_entrypoint_URL_length	8	uimsbf
SG_entrypoint_URL()	var	uimsbf
}		

SG_entrypoint_URL_length – This 8-bit field shall specify the length (in bytes) of the SG_entrypoint_URL() character string.

SG_entrypoint_URL() – This character string shall be the URL of the SG entry point.

Note: Section 11 requires the support for IPv6 for interaction channels.

8 TIME OF DAY SIGNALING

The NTP timescale shall be UTC according to Sections 4 and 6 of NTPv4 [21]. The refid field shall be set to one of the UTC sources as defined in Figure 12 of NTPv4 [21]. NTP packets shall be sent at least once every 10 minutes in every Ensemble in the M/H Broadcast using multicast server mode on the IANA assigned addresses and port. Note these assigned numbers are port 123, IPv4 multicast address 224.0.1.1. See NTPv4 [21] Section 8.

9 FILE DELIVERY

This section describes the methods used to deliver files over an ATSC M/H physical link to an ATSC M/H compliant terminal device.

In general, there are two types of files that might be delivered using the methods described in this Part. The first of these is content files, such as music or video files. The second type of file that may be transmitted is a portion of the service guide. This includes long- and short-term keys for service protection, logos and SDP files. In either case, the delivery mechanisms are the same and it is up to the terminal to resolve the purpose of the files.

FLUTE [16] generally refers to a set of protocols, including ALC [34] and LCT [35], which provide the ability to deliver a named file over a unidirectional channel using UDP [24] datagrams. Section 6.3 describes how UDP packets can be delivered over an ATSC M/H physical channel.

The methods referenced include some description of the content using the FDT and SDP [32] [15] files. However, most of the content description is carried over other protocols. For example, content files delivered to a terminal would normally be described within the Service Guide.

Similarly, the set of file types and formats that are acceptable is beyond the scope of this document.

The set of protocols collectively described as FLUTE include numerous options. To maximize compatibility with other worldwide standards, the ATSC-M/H standard restricts the usage of FLUTE in exactly the same fashion as the OMA BCAST family of standards.

In general, these restrictions are designed to reduce the number of operating points to ensure maximal compatibility and simplify receiver design. Some examples of the restrictions imposed by OMA BCAST include:

- A number of FEC modes are possible according to the FLUTE standard [36]. OMA BCAST only requires the terminal to support the No FEC mode of operation.
- The length of the CCI field is extensible according to the base FLUTE standard. OMA BCAST restricts this field to 32-bits only and requires it to be set to 0.

Furthermore, this standard does not define the usage of ALC (without FLUTE) for file delivery over ATSC M/H as it is under-specified.

In addition to specifying the usage of the FLUTE protocol, the OMA BCAST standard also specifies the syntax for advertising the details of the FLUTE session using the SDP protocol.

The delivery of files over ATSC-M/H channels conforms to the OMA BCAST specifications for file delivery. The requirements for file delivery as specified by OMA BCAST are detailed in Section 5 of the OMA BCAST File and Stream Distribution. Section 5.4, "File Distribution over back-end interfaces" is not applicable.

The file repair procedure is used for subsequent repair of an incomplete or corrupted file reception. The file repair procedure allows a receiver to connect to a file repair server and request missing packets (or encoding symbols), source blocks, or complete files.

The file repair procedure conforms to the one defined in OMA BCAST Distribution [22] Section 5.3.3.

The description of the file repair procedure is given in the "associatedProcedureDescription" file, which is delivered to receivers according to OMA BCAST Distribution [22] Section 5.3.1.

The reception reporting procedure allows the service provide to collect statistical information about the reception status of the delivered files.

The reception reporting procedure conforms to the one defined in OMA BCAST Distribution [22] Section 5.3.2.

The description of the reception reporting procedure is given in the "associatedProcedureDescription" file, which is delivered to receivers according to OMA BCAST Distribution [22] Section 5.3.1.

9.1 System Specifications (Normative)

The transmission of files shall conform to OMA BCAST File and Stream Distribution for Mobile Broadcast Services [22], except Section 5.4 and Section 6.

9.1.1 FDT Schema Extensions

In addition to specifying the usage of the FLUTE protocol, the OMA BCAST standard also specifies the syntax for describing the details of the FLUTE session using the SDP protocol. It also gives a valid and extended schema for the FDT in Section 5.2.6.6.1. In this standard, no bearer specific restrictions or extensions apply and as such, the type of the FDT schema shall be the default type (FDT-InstanceType).

10 STREAMING DELIVERY

The Service Multiplex and Transport Subsystem for ATSC-M/H services enables the encapsulation of streaming media in layers that are interoperable with today's standard Internet

Protocol [5] data network. The goal is to enable data flowing over Internet Protocol [5] data network to treat the ATSC-M/H transmission as a seamless extension of such a network.

This section describes the technique for delivering steaming real-time media to an ATSC-M/H reception device. Other delivery techniques may be used for the delivery of non-real time services.

For ATSC-M/H the Real-time Transport Protocol [23] usage is defined for real-time streaming media sessions to provide both timing and multiplexing information. For ATSC-M/H the RTP layer is carried using User Datagram Protocol (UDP) [24] Layer. Since there are no error correction facilities at the Transport or Session Layers, it is assumed that the physical layer will have error correction capabilities and that the media codec layer will be able to tolerate minimal amounts of data corruption. It is expected that total loss of a significant number of RTP packets can occur at a random and undeterminable rate. This is expected when a receiver is operating near the signal reception threshold. Logically the end of each such outage period may be considered to be equivalent to initial service acquisition. Receiver behavior and feedback to the consumer during periods where data is not provided from the physical layer is not defined, but is expected to be gracefully handled.

In the ATSC-M/H environment RTP provides such functionality as stream identification, packet loss detection, presentation timing for media elements, stream synchronization information, and further data framing. As well this Part of the ATSC-M/H Standard specifies a buffer model to insure a stricter timing model than typically available in conventional RTP.

RTP has two components: 1) a set of features for data delivery, and 2) a set of features for media control (RTCP). These features are sent as two different streams of data on adjacent UDP ports.

The ATSC-M/H buffer model provides for a mechanism to insure that the time base of the media creation and delivery system is able to be recreated by the client device. This is intended to enable both quality stream synchronization (A/V synchronization) as well as long term playback without buffer underflow or overflow.

10.1 UDP for Streaming Media Transport Layer

When streaming real-time media, the Transport layer of the ATSC-M/H system shall utilize the Real-time Transport Protocol [23] encapsulated by User Datagram Protocol (UDP) [24]. UDP is inherently defined as unreliable transfer. Thus, the higher level layers in the ATSC-M/H network stack are unaware if data is lost. UDP does provide a 16 bit checksum which can indicate if the contents of the UDP datagram are in-tact or corrupt. User Datagram Protocol also provides source and destination port numbers.

10.1.1 UDP Constraints in ATSC-M/H

10.1.1.1 UDP Port Range

ATSC-M/H shall conform to Internet port use policy as defined in RFC 4340 Section 19.9 [25]. More information and registration can be found at IANA [37].

10.1.1.2 UDP Packet Length

In ATSC-M/H the maximum allowable length in the UDP length field shall be as defined in Section 6.3 of this document.

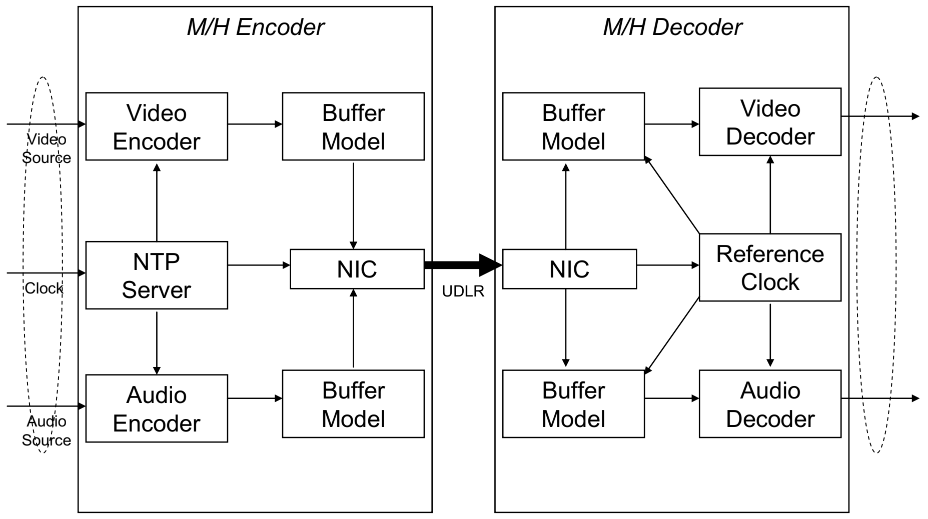

Figure 10.1 End-to-end architecture.

10.2 RTP for Streaming Media Session Layer

All streaming media shall utilize the Real-time Transport Protocol (RTP) [23], which shall be carried in UDP/IP. The requirements of RTP streaming shall be set forth in Section 6.2 (not including 6.2.1 or 6.2.2 or 6.2.3) of File and Stream Distribution for Mobile Broadcast Services [22].

In addition the following requirements shall be met:

- RTP packets shall always be on an even port and the RTCP packets shall always be on the next port number in sequence (an odd port).
- For the streaming of real-time media in ATSC-M/H RTP Header extensions shall not be present.
- To decrease protocol overhead several media frames can be encapsulated into a single RTP packet. All such encapsulated media frames are defined in the media-specific encapsulations. See A/153 Part 7 for AVC [38] and A/153 Part 8 for HE AACv2 [39].

10.3 Timing and Buffer Model for Streaming Media

All ATSC-M/H encoder devices shall adhere to the timing and buffer models defined in this section when IP packets are used.

10.3.1 Introduction

This section provides timing and buffer models for multiple RTP streams over a unidirectional link route (UDLR), as well as remote contributions over multiple links. The intent is to enable a model and timing mechanism with equivalent synchronous properties to the buffer and timing model of MPEG-2 Transport Streams (TS) as defined in ISO/IEC 13818-1 [18].

The premise of the time base and buffer model design is that the MPEG-2 TS models are both adequate and necessary to achieve proper sync and reasonable acquisition times for video/audio services. Hence, this section maps those MPEG-2 TS models onto an NTP/RTP-based delivery system. The end-to-end architecture of the system is shown in Figure 10.1.

The video and audio encoder boxes are the compression algorithm plus any conversion of the source material, and any buffering adjustment needed for preserving video and audio sync on their outputs (i.e., delay in the audio encoder to account for the traditionally larger video encoder delay). The encoders are also responsible for maintaining the elementary stream buffer models, which are outside the scope of this document. The encoder buffer models are the transport layer models that regulate the output of video and audio RTP/UDP/IP packets. The NTP server generates a stable reference timebase according to NTP [26] and NTPv4 [21]. The Network Interface Controller (NIC) is the hardware interface to the physical transport layer. In the case of the encoder, this includes the last point in the transport creation where the position of each IP packet in time is known and fixed. In the case of ATSC-M/H, this is not necessarily the output of the encoder device, but rather the device that is encoding the packets onto the physical layer RS Frames. And, it may be a collection of different physical devices after the encoder and before the final emission.

The encoder NIC is responsible for re-time-stamping all timebase values as the packet output location is determined. The NIC is conceptually all physical devices that affect the position of IP packets in the emission after the compression system.

The decoder reference clock is a reconstruction of the decoder timebase and provides various corrections for jitter and discontinuities. The decoder buffer models are a simplified version of the buffer model that extracts pictures at prescribed times. The video and audio decoders decode the elementary streams to rendering hardware, and include any delay needed to preserve the video audio sync (i.e., audio to account for the decode delays).

10.3.2 Timing Model

In order for RTP streams to be synchronized and to enable a buffer model, a specific timing model is required. The timing model consists of:

- An encoder clock in the encoder clock timebase
- A transport timestamp
- Access unit presentation times
- A decoder clock

The encoder clock shall be a monotonic synchronous physical clock.

The transport timestamp shall be the NTP transmit timestamp field as defined in NTP [26]. Each media service shall contain a single transport timestamp stream. The timebase as signaled by the reference_ID field should be "PPS", but may be a time of day, such as "GPS". See NTPv4 [21].

The access unit presentation times are a function of the RTP timestamp field and the RTCP SR NTP timestamp and RTP timestamp mapping, and the result is the MPEG-2 TS equivalent of the Presentation Time Stamp (PTS). Each media component stream shall contain a RTCP SR stream. The decoder clock is the NTP sys.clock parameter and is the equivalent of the MPEG-2 TS System Time Clock (STC). The PTS in terms of the decoder STC is derived as follows:

$$\text{PTS} = \text{RTCP\_SR\_NTP\_timestamp} + (\text{RTP\_timestamp} - \text{RTCP\_SR\_RTP\_timestamp}) / \text{media\_clock\_rate}$$

NTP and RTCP SR packets shall be sent periodically, and should be sent within 100 ms following every video random access point and every audio frame in order to minimize tuning and synchronization delays.

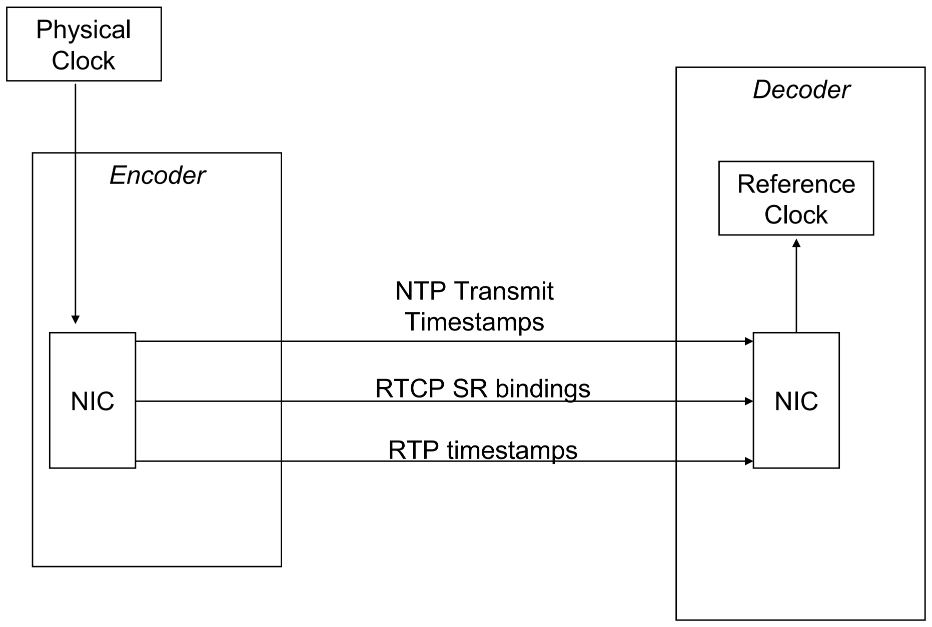

Figure 10.2.Time model elements.

Note: Due to the bursty nature of the M/H RS frame emission, one NTP timestamp per media service and one RTCP SR record per media component stream per RS frame will meet the above timing requirement.

The above time model elements are shown graphically in Figure 10.2.

The encoder clock shall meet the following constraints:

- Resolution of at least 1.11x10^{-5} seconds (i.e., 90 kHz)
- Jitter not to exceed 5.0x10^{-7} seconds (i.e., 500 ns)
- Drift not to exceed 7.5x10^{-2} Hz/second
- Monotonically increasing
- No discontinuities within a content stream, for the duration of the stream (although rollover is permitted)

The video RTP timestamp field should be 90 kHz and the audio RTP timestamp field should be set to the audio sample rate.

The decoder clock should have the same specifications as the encoder clock. The decoder clock should stay in sync with the transport timestamp encoded in the transport according to the techniques defined in NTP [26], and particularly NTPv4 [21].

In order to provide an accurate and nearly jitter-free encoder reference clock in the transport, the NIC (collectively all the devices that modify the IP packet location in the transport) shall be encoder clock and NTP “aware”. After power up, the first NTP packet should be transmitted as close to actual encoder time as possible.. After that, the NIC shall keep the NTP transmit timestamp accurate with respect to the encoder timebase. Any repositioning of the NTP encoder clock packets within the transmission stream shall require an adjustment to the NTP transmit timestamp in the timebase of the encoder clock. Drift between the physical encoder clock and the NTP transmit timestamp based on the physical layer clock shall be managed within the tolerances stated above along with guidelines in ISO/IEC 13818-1 [18] and NTP [26] and [21].

The decoder should use the NTP timestamp, the individual RTP timestamps and the RTCP SR timestamps to properly synchronize the presentation of video and audio.

The presentation times are conceptually at the decoder output to the viewer. The buffer models assume zero decode time, which is not possible in practice of course. And, decoding video typically takes longer than decoding audio, so decoders may have to adjust its compressed and/or uncompressed buffers to ensure synchronized presentations to the viewer. This is implementation-dependent.

10.3.3 Buffer Model

In order for RTP streams to be efficiently encoded, decoded and presented to a viewer, exact definitions of byte arrival and decoding events and the times at which these occur must be defined. This is the “buffer model”. The buffer model is a conceptual model used to define these terms precisely and to model the decoding process during the encoding or verification of the RTP streams in an IP-based transport. The buffer model is dependent upon the timing model described above for proper operation.

A buffer model is primarily used to constrain the encoding process and bitstream, not constrain or define a conformant decoder. That is, the decoder is permitted to do whatever it wishes, although precisely following the buffer model is guaranteed to produce the desired results. However, a decoder may also wish to account to some degree for non-conformant bitstreams, intentionally use larger buffers than required for trick play, etc.

This buffer model described here replaces the front end elements (t(i), TB_1, TB_n, and MB_1) of the T-STD buffer model as defined in 13818-1 [18] Sections 2.4 and 2.14.3.1 for AVC video and HE AACv2 audio. Reference to PTS shall mean the presentation time derived from the RTP timestamp and the RTCP SR mapping fields; and reference to DTS shall mean a monotonically increasing time base per access unit implicitly incremented (at 1/(video framerate) in the case of video). This new buffer model front end is shown in Figure 10.3.

Note that this buffer model is concerned only with video and audio RTP streams. It may be applicable to other streams.

Where:

- RB is the assembly buffer containing RTP packets
- RBS is the size of RB in bytes
- RBX is the rate at which data is removed from RB
- t(i) is the byte stream of a single M/H ensemble
- TB is the buffer containing IP packets
- TBS is the size of TB in bytes
- TBX is the rate at which data is removed from TB
- TS is a transport smoothing buffer
- TSX is the rate at which data is removed from TS
- TSS is the size of TS in bytes

EB and B (and the rest of the buffer model) shall be as defined in ISO/IEC 13818-1 [18].

The data from the M/H IP byte stream is bursty and therefore (unlike an MPEG-2 TS) does not enter the buffer model at a constant rate. The input to the buffer model t(i) is the byte stream of

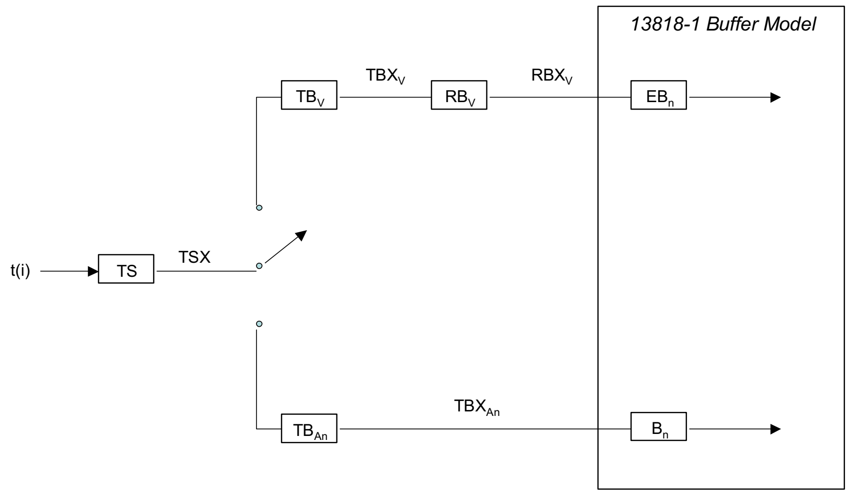

Figure 10.3 IP buffer model front end.

the physical layer for a single M/H ensemble. TS is instantaneously filled with up to TSS bytes of IP packets from the M/H ensemble every M/H Frame time. TSS shall be 187 x (N – 2) where N is defined in Equation 5.2 of A/153 Part 2 [31]. The maximum value for TSS for the defined values of N in Part 2 is 341,088 bytes. TSX shall be TSS/(M/H Frame time), where M/H Frame time is defined in Section 5.3.1.1 of A/153 Part 2 [31]. The maximum value for TSX for the largest calculated value of TSS is 352,407 bytes. TS shall not overflow. TS may underflow.

The outputs of TS are Internet Protocol (IP) packet bytes conforming to STD05 [5]. The physical layer containers and bindings are discarded prior to this point. The IP packet Total Length field is used for implied packet framing without fill.

Based on locally defined filtering criteria (e.g., IGMP-like function using destination address and port and possibly source address) IP packet bytes are delivered into a target transport buffer, TB as shown on the left side of Figure 10.3. IP packets not matching any filtering criteria are discarded. TBS shall be 2.5 x MTU[6]. TB shall not overflow, but may underflow.

IP packets are transferred from TB to RB at rate TBX and assembled into RTP packets. TBX, RBS, and RBX are stream type dependent (see the following sections). Duplicate RTP packets (based on sequence number) shall be discarded. RB shall not overflow, but may underflow.

10.3.4 AVC

AVC RTP streams are required to conform to RFC 3984 by Part 7 of this Standard. The RTP streams shall conform to the buffer model front end defined above and as defined in ISO/IEC 13818-1 [18], Section 2.14.3.1.

TBX shall be as defined in ISO/IEC 13818-1 [18] Section 2.14.3.1 for Rx_n.

RBS shall be as defined in ISO/IEC 13818-1 [18] Section 2.14.3.1 for MBS_n.

RBX shall be as defined in ISO/IEC 13818-1 [18] Section 2.14.3.1 for Rbx_n.

6. This is roughly equivalent to the MPEG-2 TS fixed size of 512 bytes for 188-byte TS packets.

10.3.5 HE AACv2

HE AACv2 RTP streams are required to conform to RFC 3640 by Part 8 of this Standard. RTP streams shall conform to the buffer model front end defined above and to the general audio buffer model as defined in ISO/IEC ISO/IEC 13818-1 [18], Section 2.4.2.

TBX shall be as defined in ISO/IEC 13818-1 [18] Section 2.4.2.3 for Rx_n for “other audio” ($2.6x10^6$ bps).

11 INTERACTION CHANNEL

The presence of a receiver interaction channel is optional. When present, it shall conform to ATSC A/96 [27], Sections 6 and 7. Other protocols may be supported, including those defined in Sections 8 and 9.

IPv4 and IPv6 addressing shall both be supported.

Note: When Service Protection [33] is present in combination with an interaction channel, other protocols might be required.

Note: When the Application Framework [40] is present in combination with an interaction channel, other protocols might be required.

Device provisioning is not defined here, but shall meet the requirements of A/96 [27], Section 7.3.

Annex A: Error Indicator in the M/H TP Headers (Informative)

For transmission robustness, each RS Frame is delivered with RS encoding for each of the N-columns, introducing P bytes of RS parity per each column (where the value of P is determined by the RS Code Mode), and a two-byte CRC syndrome checksum at the end of each row, as shown in Figure A1.

Upon the reception of an RS Frame, the payload of the RS Frame (187 M/H TPs) is recovered by following a three-step process, utilizing the RS parity bytes and the CRC Checksum bytes, as described below.

1) An error detection procedure, utilizing the CRC Checksum at the end of every row of the RS Frame, is performed on each row (thus on each M/H-TP and the CRC Checksum itself) of the RS Frame. If the result of the CRC-check reports an error occurrence(s) in a row, all the bytes of that row are marked with the error flag. The purpose of this marking is to enable the "Erasure" RS decoding in step 2.
2) An error correction procedure, utilizing the RS parity, is performed on each column of the RS Frame except the last two columns (CRC Checksum columns), utilizing the error flags reported by CRC-check in step 1. In this step, the "Erasure" RS decoding corrects the bytes

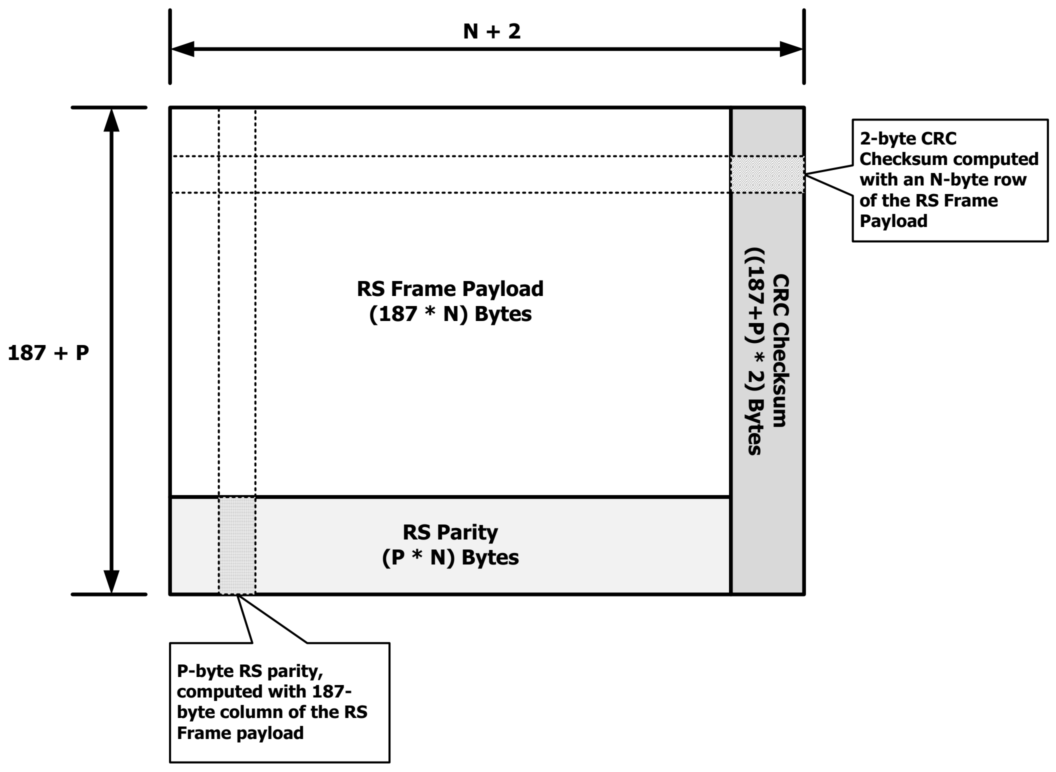
N + 2
187 + P
RS Frame Payload
(187 * N) Bytes
RS Parity
(P * N) Bytes
CRC Checksum
((187+P) * 2) Bytes
2-byte CRC Checksum computed with an N-byte row of the RS Frame Payload
P-byte RS parity, computed with 187-byte column of the RS Frame payload

Figure A1 RS Frame with RS-CRC encoding.

containing the error occurrences in the RS Frame payload. Along with the error correction, the RS decoding procedure reports whether all the error occurrences in the RS Frame payload have been corrected with the RS code capability or there still are some error occurrences that could not be corrected. If the RS decoding reports that all the bytes containing errors have been corrected in the RS Frame payload, the error_indicator fields of all the M/H TPs are set to '0'.

3) A repeat of the error detection procedure, utilizing the CRC Checksum at the end of every row of the RS Frame, if the report of the RS decoding procedure says there still are error occurrences that could not be corrected by RS decoding. If the result of the CRC-check still reports an error occurrence in a row, then the error_indicator of the M/H TP corresponding to that row is set to '1'.

However, error occurrences in the CRC Checksum bytes themselves are neither identified during the error detection procedures nor corrected by the error correction procedure. Thus, when the error_indicator field is set to '1' in an M/H TP, there is a chance that the error_indicator is set due to an error occurrences in the CRC Checksum bytes, rather than error occurrences in the M/H TP itself. However, when the error_indicator field is set to '1' in an M/H TP, the probability of having the error occurrences in the M/H TP that could not be corrected is still much higher than the probability of having the error occurrences in the CRC Checksum, it is recommended to discard the M/H TPs that have error_indicator field set to '1'.

Annex B: Allocation of MH_service_id Numbers (Normative)

B1 OVERVIEW

In order to support regional M/H Services, for which it is desired to have a single MH_service_id for all instances of a service, regardless of the transmitter from which the instance of the service is broadcast, the set of possible values for the MH_service_id shall be divided into ranges based on the value of the high order byte of the MH_service_id, as follows:

- Value of high order byte in the range 0 – 1 (values 0 – 511):
  Reserved by ATSC for special uses, such as perhaps EAS services
- Value of high order byte in the range 2 – 69 (values 512 – 17919):
  Reserved for local M/H Services
- Value of high order byte in the range 70 – 255 (values 17920 – 65535):
  Reserved for regional M/H Services

A "local M/H Service" is a service for which the broadcaster of the service intends for the MH_service_id to be unique only within the local broadcast area.

Allocation of the MH_service_ids reserved for local M/H Services shall be allocated among local broadcasters in each local area according to the value of the high order byte, using exactly the same rules as those specified in Annex B of ATSC A/65 [3] for the allocation of major channel numbers in the range from 2 to 69.

A "regional M/H Service" is a service for which the content provider and/or service provider intend to have instances of the service broadcast in more than one local broadcast area, with the same MH Service ID for all the instances.

Allocation of the MH_service_ids reserved for regional services shall be allocated via a registration authority, so that they are unique throughout the region (where a typical "region" would be a country).

advancedtelevisionsystemscommitteeinc

Advanced Television Systems Committee, Inc.
1776 K Street, N.W., Suite 200
Washington, D.C. 20006