ATSC Standard: Link-Layer Protocol

Doc. A/330:2023-03
28 March 2023

Advanced Television Systems Committee
1300 I Street, N.W., Suite 400E
Washington, D.C. 20005
202-872-9160

The Advanced Television Systems Committee, Inc. is an international, non-profit organization developing voluntary standards and recommended practices for broadcast television and multimedia data distribution. ATSC member organizations represent the broadcast, professional equipment, motion picture, consumer electronics, computer, cable, satellite, and semiconductor industries. ATSC also develops implementation strategies and supports educational activities on ATSC standards. ATSC was formed in 1983 by the member organizations of the Joint Committee on Inter-society Coordination (JCIC): the Consumer Technology Association (CTA), the Institute of Electrical and Electronics Engineers (IEEE), the National Association of Broadcasters (NAB), the Internet & Television Association (NCTA), and the Society of Motion Picture and Television Engineers (SMPTE). For more information visit www.atsc.org.

Note: The user's attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights. By publication of this standard, no position is taken with respect to the validity of this claim or of any patent rights in connection therewith. One or more patent holders have, however, filed a statement regarding the terms on which such patent holder(s) may be willing to grant a license under these rights to individuals or entities desiring to obtain such a license. Details may be obtained from the ATSC Secretary and the patent holder.

Implementers with feedback, comments, or potential bug reports relating to this document may contact ATSC at https://www.atsc.org/feedback/.

Revision History

Version	Date
Candidate Standard approved	25 December 2015
A/330:2016 Standard approved	19 September 2016
A/330:2019 Standard approved	03 May 2019
Typo fixed in Section 7.1, second paragraph; corrected port number to read “4937/udp”.	20 June 2019
A/330:2022-03 (references to ATSC documents updated)	31 March 2022
Corrected broken hyperlink on page 29	25 January 2023
A/330:2023-03 (references to ATSC documents updated)	28 March 2023

Table of Contents

Index of Figures

Index of Tables

ATSC Standard: Link-Layer Protocol

1. SCOPE

This Standard defines the ATSC Link-layer Protocol (ALP). ALP corresponds to the data link layer in the OSI 7-layer model. ALP provides a path to deliver IP packets, link layer signaling packets, and MPEG-2 Transport Stream (TS) packets down to the RF Layer and back, after reception. ALP also optimizes the proportion of useful data in the ATSC 3.0 Physical Layer by means of efficient encapsulation and overhead reduction mechanisms for IP and MPEG-2 TS transport. ALP provides extensible headroom for future use.

This Standard consists of the following functions:

- ALP Packet Format
- IP header compression
- Link layer signaling

1.1 Organization

This document is organized as follows:

- Section 1 – Outlines the scope of this document and provides a general introduction.
- Section 2 – Lists references and applicable documents.
- Section 3 – Provides a definition of terms, acronyms, and abbreviations for this document.
- Section 4 – Link Layer Overview
- Section 5 – ALP Packet Format
- Section 6 – IP Header Compression
- Section 7 – Link Layer Signaling
- Annex A – ALP System Architecture
- Annex B – ALP Packet Format Examples
- Annex C – Compressed IP Packet
- Annex D – IP Header Compression Examples
- Annex E – ROHC Synchronization

2. REFERENCES

All referenced documents are subject to revision. Users of this Standard are cautioned that newer editions might or might not be compatible.

2.1 Normative References

The following documents, in whole or in part, as referenced in this document, contain specific provisions that are to be followed strictly in order to implement a provision of this Standard.

[1] ATSC: “ATSC Standard: Physical Layer Protocol,” Doc. A/322:2023-03, Advanced Television Systems Committee, Washington, DC, 28 March 2023.

[2] ATSC: “ATSC Standard: Signaling, Delivery, Synchronization, and Error Protection,” Doc. A/331:2023-03, Advanced Television Systems Committee, Washington, DC, 28 March 2023.

[3] IEEE: “Use of the International Systems of Units (SI): The Modern Metric System,” Doc. SI 10, Institute of Electrical and Electronics Engineers, New York, NY.

[4] IETF: "Internet Protocol," Doc. STD05 (originally RFC 791), Internet Engineering Task Force, Reston, VA, September 1981.
[5] IETF: "User Datagram Protocol," Doc. STD06 (originally RFC 768), Internet Engineering Task Force, Reston, VA, August 1980.
[6] ISO/IEC: 13818-1:2013(E), "Information technology – Generic coding of moving pictures and associated audio information: Systems – Part 1."
[7] IETF: RFC 3095, "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed", Internet Engineering Task Force, Reston, VA, July 2001. http://tools.ietf.org/html/rfc3095
[8] IETF: RFC 4815, "RObust Header Compression (ROHC): Corrections and Clarifications to RFC 3095", Internet Engineering Task Force, Reston, VA, February 2007. http://tools.ietf.org/html/rfc4815
[9] IETF: RFC 5795, "The RObust Header Compression (ROHC) Framework", Internet Engineering Task Force, Reston, VA, March 2010. http://tools.ietf.org/html/rfc5795

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 [3] shall be used. Where an abbreviation is not covered by IEEE practice or industry practice differs from IEEE practice, the abbreviation in question will be described in Section 3.3 of this document.

3.1 Compliance Notation

This section defines compliance terms for use by this document:

shall – This word indicates specific provisions that are to be followed strictly (no deviation is permitted).

shall not – This phrase indicates specific provisions that are absolutely prohibited.

should – This word indicates that a certain course of action is preferred but not necessarily required.

should not – This phrase means a certain possibility or course of action is undesirable but not prohibited.

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 Elements

One or more reserved bits, symbols, fields, or ranges of values (i.e., elements) may be present in this document. These are used primarily to enable adding new values to a syntactical structure without altering its syntax or causing a problem with backwards compatibility, but they also can be used for other reasons.

The ATSC default value for reserved bits is '1'. There is no default value for other reserved elements. Use of reserved elements except as defined in ATSC Standards or by an industry standards setting body is not permitted. See individual element semantics for mandatory settings and any additional use constraints. As currently-reserved elements may be assigned values and meanings in future versions of this Standard, receiving devices built to this version are expected to ignore all values appearing in currently-reserved elements to avoid possible future failure to function as intended.

3.3 Acronyms and Abbreviations

The following acronyms and abbreviations are used within this document.

ALP – ATSC Link-layer Protocol

ATSC – Advanced Television Systems Committee

BSID – Broadcast Stream ID

bslbf – bit string, left bit first

CID – Context Identifier

DF – Don't Fragment (a flag in IPv4 header)

IANA – Internet Assigned Numbers Authority

IETF – Internet Engineering Task Force

IP – Internet Protocol

IR – Initialization and Refresh

IR-DYN – IR Dynamic

JSON – JavaScript Object Notation

LLS – Low Level Signaling

LMT – Link Mapping Table

LSB – Least Significant Bit

MPEG – Moving Picture Experts Group

MSB – Most Significant Bit

MTU – Maximum Transmission Unit

PDU – Protocol Data Unit

PID – Packet Identifier

PLP – Physical Layer Pipe

RAP – Random Access Point

RDT – ROHC-U Description Table

RFC – Request For Comment (IETF standard)

ROHC – RObust Header Compression

SID – Sub-stream Identifier

SLS – Service Layer Signaling

SLT – Service List Table

SN – Sequence Number

TS – Transport Stream

UDP – User Datagram Protocol

uimsbf– unsigned integer, most significant bit first

XML – eXtensible Markup Language

3.4 Terms

The following terms are used within this document.

Additional Header – An Additional Header is part of the ALP packet header. The presence of an Additional Header depends on the specific field values of the Base Header.

Base Header – A Base Header is part of the ALP packet header. A Base Header is always included in the header of an ALP packet and is the first part of an ALP packet.

Broadcast Stream – The abstraction for an RF Channel which is defined in terms of a carrier frequency centered within a specified bandwidth.

Catalog RDT – A type of RDT delivery that contains the context data for multiple ROHC/ALP streams and can be carried in a robust signaling PLP. Thus, it can supply multiple RDTs for multiple ALP streams.

Context – RFC 3095 [7] describes Context, as used herein, in the following manner: 'The context of the compressor is the state it uses to compress a header. The context of the decompressor is the state it uses to decompress a header. Either of these or the two in combination are usually referred to as "context," when it is clear which is intended. The context contains relevant information from previous headers in the packet stream, such as static fields and possible reference values for compression and decompression. Moreover, additional information describing the packet stream is also part of the context, for example information about how the IP Identifier field changes and the typical inter-packet increase in sequence numbers or timestamps.'

Discrete RDT – A type of RDT delivery that is carried exclusively within the ALP stream that it supports. This approach is consistent with RFC 3095 [7] (ROHC), wherein the context is always carried within the related IP stream.

Extension Header – An Extension Header is part of the ALP packet header. The presence of an Extension Header depends on the specific field values of the Additional Header.

IP-ID – Identification field of IPv4 packet.

multicast – (verb) to send data across (e.g., an IPv4) network to many recipients simultaneously; (noun) a set of data sent across (e.g., an IPv4) network to many recipients simultaneously.

Network Layer Packet – A Network Layer Packet is a source packet in the protocol of the data to be transported, e.g., an MPEG-2 Transport Stream packet or an Internet Protocol packet.

Null Packet – An MPEG-2 TS packet with PID equal to 0x1FFF.

Real Time Service RAP – A Real Time Service Random Access Point (Real Time Service RAP) is defined as a point in the delivery of the data required to start a Real Time Service, at which point a receiver can acquire that Service with minimal delay.

reserved – Set aside for future use by a Standard.

Service – A collection of media components presented to the user in aggregate; the components can be of multiple media types; a Service can be either continuous or intermittent; a Service can be Real Time or Non-Real Time; a Real Time Service can consist of a sequence of TV programs.

3.5 Extensibility

The protocols specified in the present standard are designed with features and mechanisms to support extensibility. Receiving devices are expected to disregard reserved values and unrecognized or unsupported table types.

4. LINK LAYER OVERVIEW

The link layer is the layer between the physical layer and the network layer. The link layer transports the data from the network layer to the physical layer at the sending side and transports the data from the physical layer to the network layer at the receiving side as shown in Figure 4.1. While Figure 4.1 shows two logical flows between the link layer and physical layer, implementations are likely to utilize a single connection. The purpose of the link layer is to abstract all input packet types into a single format for processing by the physical layer (RF), ensuring flexibility and future extensibility for as-yet-undefined input types. In addition, processing within the link layer ensures that the input data can be transmitted in an efficient manner, for example by providing options to compress redundant information in the headers of input packets. Operations including encapsulation, compression and link layer signaling are referred to as the ATSC Link-layer Protocol (ALP) and packets created using this protocol are called ALP packets.

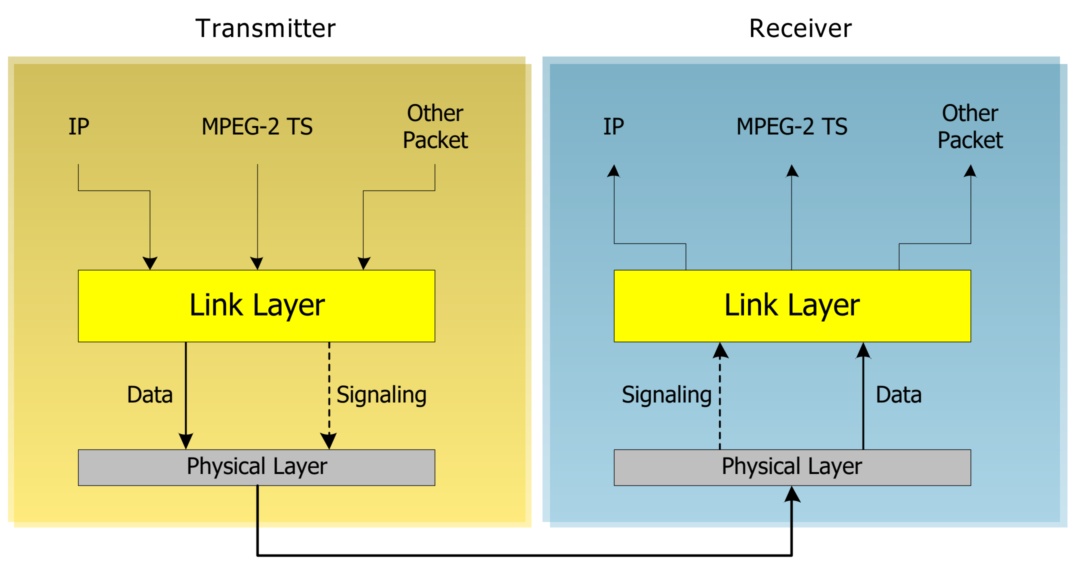

Figure 4.1 ATSC 3.0 link layer logical diagram.

4.1 Services

The services provided by ALP are briefly described below.

4.1.1 Packet Encapsulation

ALP allows encapsulation of any type of packet, including common ones such as IP [4] packets and MPEG-2 TS [6] packets. Using ALP, the physical layer need only process one single packet format, independent of the network layer protocol type (MPEG-2 TS packets are considered here as a kind of Network Layer Packet). Each Network Layer Packet or input packet is transformed into the payload of a generic ALP packet; this process is described in Section 5.1. Additionally, concatenation and segmentation can be performed in order to use the physical layer resources efficiently when the input packet sizes are particularly small or large. In this document, the term "IP packet" is used to refer to an IPv4 [4] packet unless otherwise specifically noted.

4.1.1.1 Segmentation and Reassembly

When a Network Layer Packet is too large to process easily in the physical layer, it is divided into two or more segments. The link layer packet header includes protocol fields to perform segmentation on the sending side and reassembly on the receiving side. This operation is described in Section 5.1.2.2. When the Network Layer Packet is segmented, each segment shall be encapsulated into an ALP packet and transmitted in the same order as its original position in the Network Layer Packet. Each ALP packet which includes a segment of a Network Layer Packet shall be transported within the PHY layer consecutively.

4.1.1.2 Concatenation

When a Network Layer Packet is small enough for the payload of a link layer packet to include several Network Layer Packets, the link layer packet header includes protocol fields to perform concatenation. Concatenation involves combining multiple small-sized packets into the payload of one ALP Packet. This operation is described in Section 5.1.2.3. When the Network Layer Packets are concatenated, each Network Layer Packet shall be concatenated into the payload of an ALP packet in the same order as the original input order. Also, each Network Layer Packet which constructs a payload of an ALP packet shall be a whole packet, not a packet segment.

4.1.2 Overhead Reduction

Use of ALP can result in significant reduction in overhead for transport of data on the physical layer. Two particular cases of interest are presented below: IP packets in Section 4.1.2.1 and TS packets in Section 4.1.2.2.

4.1.2.1 IP Overhead Reduction

IP packets have a fixed header format; however some of the information which is needed in a communication environment may be redundant in a broadcast environment. ALP provides mechanisms to reduce the broadcast overhead by compressing headers of IP packets including UDP [5] headers.

4.1.2.2 MPEG-2 TS Overhead Reduction

ALP provides the following overhead reduction functionality to efficiently transport MPEG-2 TS packets. First, sync byte removal provides an overhead reduction of one byte per TS packet. Secondly, a null packet deletion mechanism removes the 188-byte null TS packets in a manner such that they can be re-inserted at the receiver, and finally, a common header removal mechanism is supported.

4.1.3 Signaling Transmission

In ALP a specific format for signaling packets is provided to allow transportation of link layer signaling. This is described in Section 5.2.

4.2 System Architecture

The functional architecture block diagram and interface is shown in Figure 4.2.

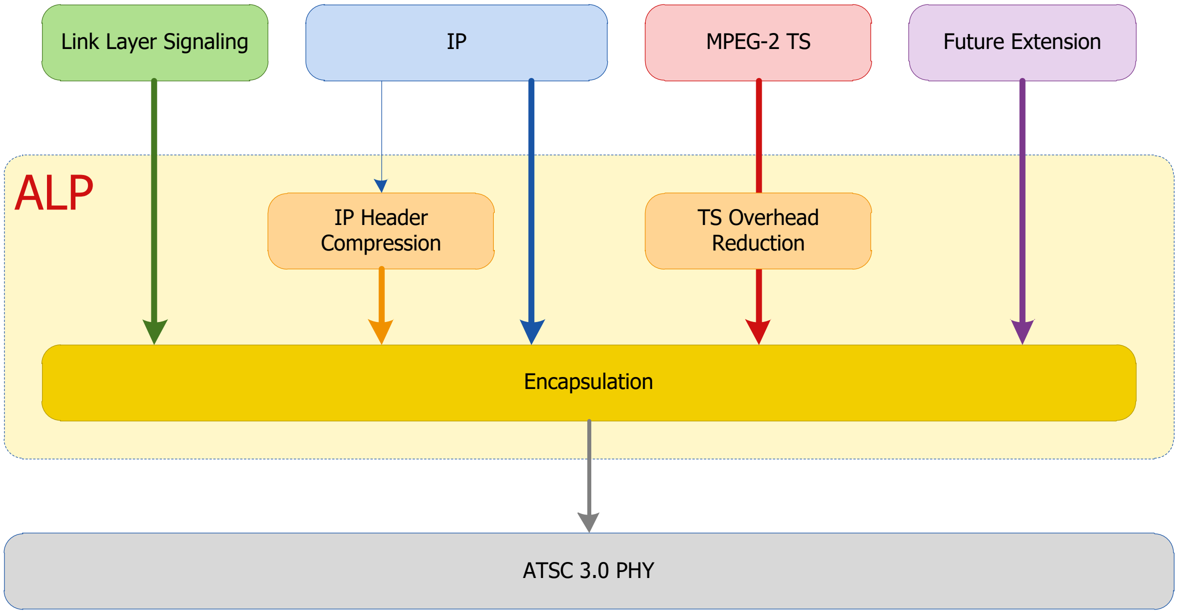

Figure 4.2 Block diagram of the architecture and interface of ALP.

ALP takes Network Layer Packets such as IPv4 and MPEG-2 TS as input packets. Future extensions might permit other packet types and protocols as inputs to ALP. ALP also specifies the format and signaling for any link layer signaling, including information about mapping specific IP packet streams to data pipes in the physical layer. Figure 4.2 also shows how ALP incorporates mechanisms to improve the efficiency of transmission, via various header compression and deletion algorithms.

5. ALP PACKET FORMAT

An ALP packet shall consist of a header followed by a data payload. The header of an ALP packet shall always contain a Base Header, and may contain an Additional Header depending on the control fields of the Base Header. The presence of an optional Extension Header is indicated by flag fields within the Additional Header. (See Figure 5.1.)

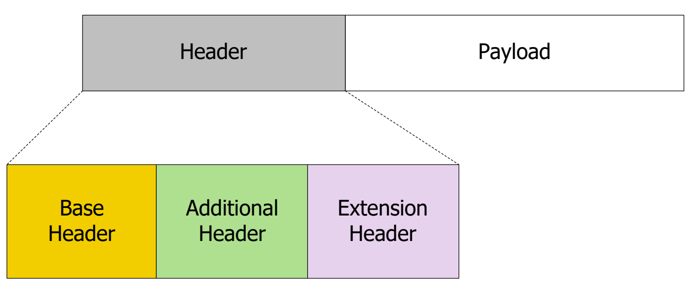

Figure 5.1 ALP Packet format.

5.1 ALP packet Encapsulation

5.1.1 Base Header

The Base Header for ALP packet encapsulation has the hierarchical structure shown in Figure 5.2. The Base Header shall always be two bytes in length and two bytes is consequently the minimum length of the ALP packet header. An exception is when the value of packet_type is '111'. In this case, the detailed structure of encapsulation is described in Section 5.4.

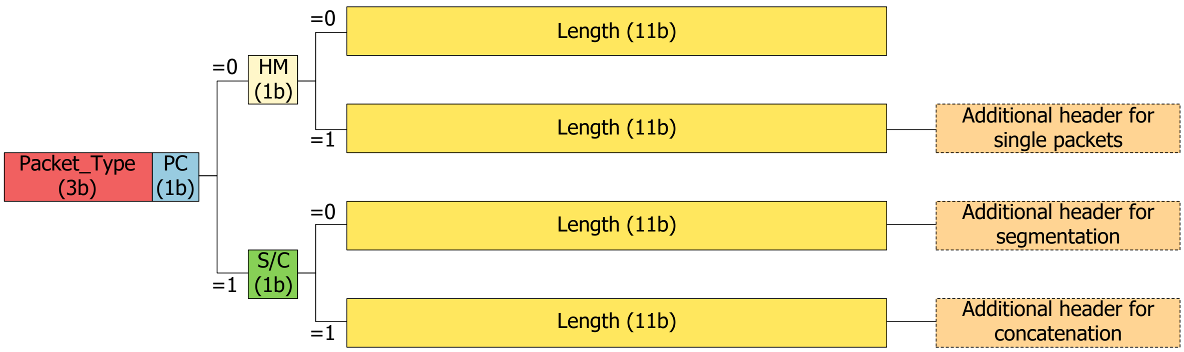

Figure 5.2 Structure of Base Header for ALP packet encapsulation.

The bit stream syntax of an ALP packet shall be as shown in Table 5.1. The text following Table 5.1 describes the semantics of each field in the table section.

Table 5.1 Header Syntax for ALP Packet Encapsulation

Syntax	No. of bits	Format
ALP_packet_header() {		
packet_type	3	uimsbf
payload_configuration	1	bslbf
if (payload_configuration == 0) {		
header_mode	1	bslbf
length	11	uimsbf
if (header_mode == 1) {		
single_packet_hdr()	var	Sec. 5.1.2.1
}		
}		
else if (payload_config == 1) {		
segmentation_concatenation	1	bslbf
length	11	uimsbf
if (segmentation_concatenation == 0) {		
segmentation_hdr()	var	Sec. 5.1.2.2
}		
else if (segmentation_concatenation == 1) {		
concatenation_hdr()	var	Sec. 5.1.2.3
}		
}		
}		

packet_type – This 3-bit field shall indicate the packet type of the input data before encapsulation into this ALP packet as shown in Table 5.2 below. Values of packet_type in the range '000' to '110' shall indicate that this ALP packet is encapsulated according to the syntax shown in Table 5.1. When set to '111', i.e. MPEG-2 TS packets are encapsulated, packet_type shall indicate that this ALP packet is encapsulated according to the syntax described in Section 5.4.

Table 5.2 Code Values for packet_type

packet_type Value	Meaning
000	IPv4 packet
001	Reserved
010	Compressed IP packet
011	Reserved
100	Link layer signaling packet
101	Reserved
110	Packet Type Extension
111	MPEG-2 Transport Stream

payload_configuration (PC) – This 1-bit field shall indicate the configuration of the payload. A value of '0' shall indicate that the ALP packet carries a single, whole input packet and the following field is the header_mode field. A value of '1' shall indicate that the ALP packet carries more than one input packet (concatenation) or a part of a large input packet (segmentation) and the following field is the segmentation_concatenation field.

header_mode (HM) – This 1-bit field, when set to '0', shall indicate there is no Additional Header for the single packet as defined in Section 5.1.2.1, and that the length of the payload of the ALP packet is less than 2048 bytes. A value of '1' shall indicate that an Additional Header for the single packet as defined in Section 5.1.2.1 is present following the length field. In this case, the length of the payload is larger than 2047 bytes and/or optional features can be used (sub-stream identification, header extension, etc.). This field shall be present only when the payload_configuration field of the ALP packet has a value of '0'.

segmentation_concatenation (S/C) – This 1-bit field, when set to '0', shall indicate that the payload carries a segment of an input packet and that an Additional Header for segmentation as defined in Section 5.1.2.2 is present following the length field. A value of '1' shall indicate that the payload carries more than one complete input packet and that an Additional Header for concatenation as defined in Section 5.1.2.3 is present following the length field. This field shall be present only when the value of payload_configuration field of the ALP packet is '1'.

length – This 11-bit field shall indicate the 11 least significant bits (LSBs) of the length in bytes of payload carried by the ALP packet. When there is a length_MSB field in the following Additional Header, the length field is concatenated with the length_MSB field, and represents the LSB value portion of the actual payload length.

Four types of packet configurations are thus possible: a single packet without any Additional Header, a single packet with an Additional Header, a segmented packet, and finally a concatenated packet. The header length of an ALP packet is indicated separately with the combination of the fields in the Base Header and the Additional Header. These four types of packet configurations are shown with the total header length in Table 5.3.

Table 5.3 Payload Configuration (PC) Field Value and Total Header Length

PC Field Value	Meaning	Next Field		Additional Header Size	Additional Header Field	Total Header Length (excluding optional header)
		Name	**Value**			
0	Single packet	HM	0	-	-	2 bytes
		HM	1	1 byte	length_MSB	3 bytes
1	Segmentation or Concatenation	S/C	0	1 byte	seg_SN, LSI	3 bytes
		S/C	1	$1 + \lceil (\text{count}+1) \times 1.5 \rceil$ bytes[1]	length_MSB, count, component_length	$3 + \lceil (\text{count}+1) \times 1.5 \rceil$ bytes[1]

[1] The operation ⌈ ⌉ above is the ceiling function.

In ALP, null packets (which have zero-length payloads) shall not be generated. Therefore, the length field for single packet encapsulation with Base Header only, concatenated length indication with length and length_MSB field, and component_length field shall not have a value equal to ‘0’.

5.1.2 Additional Headers

There are three different types of Additional Headers that need to be described: the Additional Header for single packets is described in Section 5.1.2.1, the Additional Header for segmentation is described in Section 5.1.2.2 and the Additional Header for concatenation is described in Section 5.1.2.3.

5.1.2.1 Single Packets

The Additional Header for single packets shall be present only when header_mode (HM) = ‘1’. The header_mode (HM) shall be set to ‘1’ when the length of the payload of the ALP packet is larger than 2047 bytes and/or to signal additional functionality in the single packet case. The Additional Header for single packets is shown in Figure 5.3.

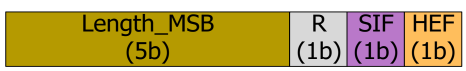

Figure 5.3 Structure of the Additional Header for single packets.

The syntax of the Additional Header for single packets shall be as shown in Table 5.4. The text following Table 5.4 describes the semantics of each field in the table section.

Table 5.4 Syntax of Additional Header for Single Packet

Syntax	No. of bits	Format
single_packet_hdr() {		
length_MSB	5	uimsbf
reserved	1	‘1’
SIF	1	bslbf
HEF	1	bslbf
}		

length_MSB – This 5-bit field shall indicate the most significant bits (MSBs) of the total payload length in bytes in the current ALP packet, and is concatenated with the length field containing the 11 least significant bits (LSBs) to obtain the total payload length. The maximum length of the payload that can be signaled is therefore 65,535 bytes. When the payload of an ALP packet is less than 2048 bytes and the optional header needs to be added to the ALP packet, this field shall be set to all zeroes.

SIF (Sub-stream Identifier Flag) – This 1-bit field shall indicate whether the optional header for sub-stream identification is present after the HEF field or not. When there is no sub_stream_identification() in this ALP packet, SIF field shall be set to '0'. When there is a sub_stream_identification() after the Additional Header in the ALP packet, the SIF shall be set to '1'. The details of the optional header for sub-stream identification are described in Section 5.1.3.1.

HEF (Header Extension Flag) – This 1-bit field shall indicate, when set to '1', that the optional header_extension() is present after the Additional Header for future extensions of the ALP header. A value of '0' shall indicate that the header_extension() is not present.

5.1.2.2 Segmentation

The Additional Header for segmentation shall be present only when segmentation_concatenation (S/C) = '0'. The Additional Header for segmentation is shown in Figure 5.4.

Figure 5.4 Structure of the Additional Header for segmentation.

The syntax of the Additional Header for segmentation shall be as shown in Table 5.5. The text following Table 5.5 describes the semantics of each field in the table section.

Table 5.5 Syntax of the Additional Header for Segmentation

Syntax	No. of bits	Format
segmentation_hdr() {		
segment_sequence_number	5	uimsbf
last_segment_indicator	1	bslbf
SIF	1	bslbf
HEF	1	bslbf
}		

segment_sequence_number – This 5-bit unsigned integer shall indicate the order of the corresponding segment carried by the ALP packet. For the ALP packet which carries the first segment of an input packet, the value of this field shall be set to '0x0'. This field shall be incremented by one for each additional segment belonging to the segmented input packet. When segment_sequence_number is equal to '0x0', last_segment_indicator shall not be equal to '1'.

last_segment_indicator (LSI) – This 1-bit field shall indicate, when set to '1', that the segment in this payload is the last segment of the input packet. A value of '0' shall indicate that it is not the last segment.

SIF (Sub-stream Identifier Flag) – This 1-bit field shall indicate whether the optional header for sub-stream identification is present after the HEF field or not. When there is no sub_stream_identification() in the ALP packet, the SIF field shall be set to '0'. When there is a sub_stream_identification() after the Additional Header in the ALP packet, the SIF field shall be set to '1'. The details of the optional header for sub-stream identification are described in Section 5.1.3.1.

HEF (Header Extension Flag) – This 1-bit field shall indicate, when set to ‘1’, that the optional header_extension() is present after the Additional Header for future extensions of the ALP header. A value of ‘0’ shall indicate that the optional header_extension() is not present.

5.1.2.3 Concatenation

This Additional Header shall be present when segmentation_concatenation (S/C) = ‘1’. The Additional Header for concatenation is shown in Figure 5.5.

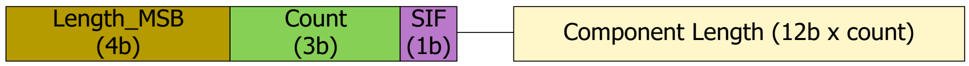

Figure 5.5 Structure of Additional Header for concatenation.

The syntax of the Additional Header for concatenation shall be as shown in Table 5.6. The text following Table 5.6 describes the semantics of each field in the table section.

Table 5.6 Syntax of Additional Header for Concatenation

Syntax	No. of bits	Format
concatenation_hdr() {		
length_MSB	4	uimsbf
count	3	uimsbf
SIF	1	bslbf
for (i=0; i<count+1; i++) {		
component_length	12	uimsbf
}		
if ((count & 1) == 0) {		
stuffing_bits	4	’0000’
}		
}		

length_MSB – This 4-bit field shall indicate the most significant bits (MSBs) of the payload length in bytes in this ALP packet. The maximum length of the payload is 32,767 bytes for concatenation.

count – This field shall indicate the number of the packets included in the ALP packet. The value of this field shall be set to (number of the packets included in the ALP packet - 2). Therefore, the minimum possible number of concatenated packets is 2 and the maximum possible number of concatenated packets is 9 in an ALP packet.

SIF (Sub-stream Identifier Flag) – This 1-bit field shall indicate whether the optional header for sub-stream identification is present after the last component_length field or not. When there is no sub_stream_identification() in the ALP packet, the SIF field shall be set to ‘0’. When there is a sub_stream_identification() after the Additional Header in the ALP packet, the SIF field shall be set to ‘1’. The details of the optional header for sub-stream identification are described in Section 5.1.3.1.

component_length – This 12-bit length field shall indicate the length in bytes of each packet. component_length fields are included in the same order as the packets present in the payload except the last component packet. The number of length fields shall be indicated by (count+1). When an ALP header contains an odd number of component_length fields, four stuffing bits shall follow after the last component_length field. These bits shall be set to ‘0’.

5.1.3 Extension Header

The ALP packet may be extended by the addition of an optional Extension Header if it is signaled by one of the fields in the Additional Header structure, that is, the SIF or HEF. The syntax of the optional Extension Header shall be as shown in Table 5.7. When the SIF flag is set, the Extension Header shall contain a sub_stream_identification() structure as described in Table 5.8. When the HEF flag is set, the Extension Header exists and shall contain a header_extension() as described in Table 5.9. When both SIF and HEF are set, the order shall be sub_stream_identification() structure followed by header_extension() structure.

Table 5.7 Syntax of Extension Header

Syntax	No. of bits	Format
extension_hdr() {		
if (SIF == 1) {		
sub_stream_identification()	8	Sec. 5.1.3.1
}		
if (HEF == 1) {		
header_extension()	var	Sec. 5.1.3.2
}		
}		

5.1.3.1 Sub-Stream Identification

The optional header for sub-stream identification is used to filter out specific packet stream(s) at the link layer level. One example of sub-stream identification is the role of service identifier in an ALP stream carrying multiple services. The mapping information between an upper layer stream and the SID value corresponding to the upper layer stream shall be provided as in the Link Mapping Table as specified in Section 7.1.1.

Table 5.8 Syntax for Sub-Stream Identification

Syntax	No. of bits	Format
sub_stream_identification() {		
SID	8	bslbf
}		

SID (Sub-stream Identifier) – This 8-bit field shall indicate the sub-stream identifier for the ALP packet. When there is an optional header_extension(), SID shall be present between the Additional Header and the optional header_extension().

5.1.3.2 Header Extension

The header extension contains extended fields for future use. Receivers are expected to ignore any header extensions that they do not understand. The header extension shall consist of the fields as defined in Table 5.9. In the present version of the specification, all values for extension_type field are reserved.

Table 5.9 Syntax for Header Extension

Syntax	No. of bits	Format
header_extension() {		
extension_type	8	uimsbf
extension_length_minus1	8	uimsbf
for (i=0; i<=extension_length_minus1; i++) {		
extension_byte	8	uimsbf
}		
}		

The semantic definitions for the fields are given below:

extension_type – This 8-bit field shall indicate the type of the header_extension() according to Table 5.10. When the value of extension_type is in the User Private range, extension_byte data is usable for proprietary implementations.

Table 5.10 Code Values for Extension Type

extension_type	Meaning
0x00–0xEF	Reserved
0xF0–0xFF	User Private

extension_length_minus1 – This 8-bit field shall indicate one less than the number of extension_byte(s) that follow.

extension_byte – A byte representing the value of the header_extension().

5.2 Signaling Encapsulation

This section provides information about how link layer signaling is incorporated into ALP packets. Signaling packets are identified by the packet_type field of the Base Header being equal to '100'.

Figure 5.6 shows the structure of the ALP packets containing an Additional Header for Signaling Information. For the signaling encapsulation, the ALP packet shall include two parts, an Additional Header for Signaling Information and the actual signaling table itself. The signaling table shall be considered as a payload of an ALP packet. Therefore, the length of the signaling table shall be indicated using length, length_MSB, and component_length fields in the ALP packet header. When an optional Extension Header is used, the optional Extension Header shall be present after the Additional Header for Signaling Information signaling_information_hdr().

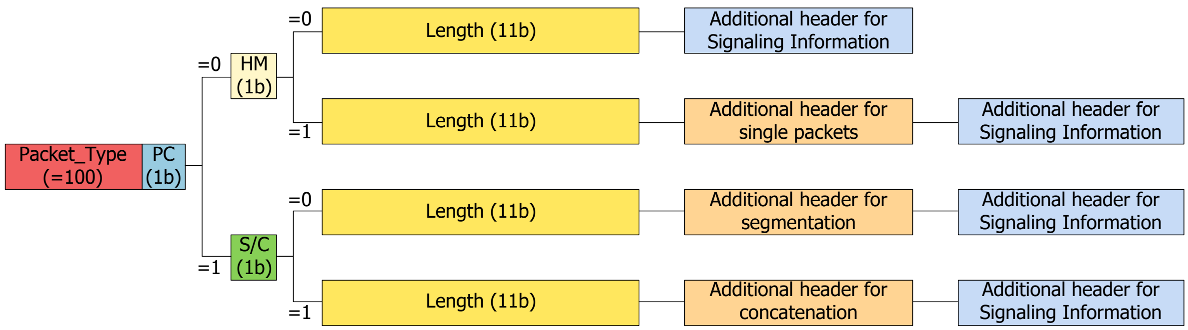

Figure 5.6 Structure of ALP signaling packets (Base Header and Additional Header).

5.2.1 Additional Header for Signaling Information

The Additional Header for Signaling Information signaling_information_hdr() shall consist of the fields as defined in Table 5.11. The text following Table 5.11 describes the semantics of each field in the table.

Table 5.11 Syntax of Additional Header for Signaling Information

Syntax	No. of bits	Format
signaling_information_hdr() {		
signaling_type	8	uimsbf
signaling_type_extension	16	bslbf
signaling_version	8	uimsbf
signaling_format	2	uimsbf
signaling_encoding	2	uimsbf
reserved	4	'1111'
}		

signaling_type – This 8-bit field shall indicate the type of signaling according to Table 5.12.

Table 5.12 Code Values for Signaling Type

signaling_type	Meaning
0x00	Reserved
0x01	Link Mapping Table (see Section 7.1.1)
0x02	ROHC-U Description Table (see Section 7.1.2)
0x03–0xEF	Reserved
0xF0–0xFF	User Private

signaling_type_extension – This 16-bit field shall indicate the attribute of the signaling. Details of this field shall be as defined in each signaling section in Section 7. This field and its value is defined within each signaling table.

signaling_version – This 8-bit field shall indicate the version of signaling. The value of this field shall be incremented by 1 whenever any data of the signaling identified by signaling_type changes. The value of signaling_version field shall wrap around to 0 after its maximum value.

signaling_format – This 2-bit field shall indicate the data format of the signaling data as described in Table 5.13.

Table 5.13 Code Values for Signaling Format

signaling_format	Meaning
00	Binary
01	XML
10	JSON
11	Reserved

signaling_encoding – This 2-bit field shall specify the encoding/compression format. The code values of signaling_encoding field are described in Table 5.14. When the signaling_format field indicates Binary (‘00’), the signaling_encoding field shall be set to ‘00’.

Table 5.14 Code Values for Signaling Encoding

signaling_encoding	Meaning
00	No Compression
01	DEFLATE (RFC 1951)
10	Reserved
11	Reserved

5.3 Packet Type Extension

In order to provide a mechanism to allow an almost unlimited number of additional protocol and packet types to be carried by ALP in the future, the Additional Header for Type Extension is defined. Packet type extension shall be used only when packet_type is ‘110’ in the Base Header as described in Table 5.2. Figure 5.7 shows the structure of ALP packets containing Additional Header for Type Extension. When an optional Extension Header is used, the optional Extension Header shall be present after the Additional Header for Type Extension type_extension_hdr().

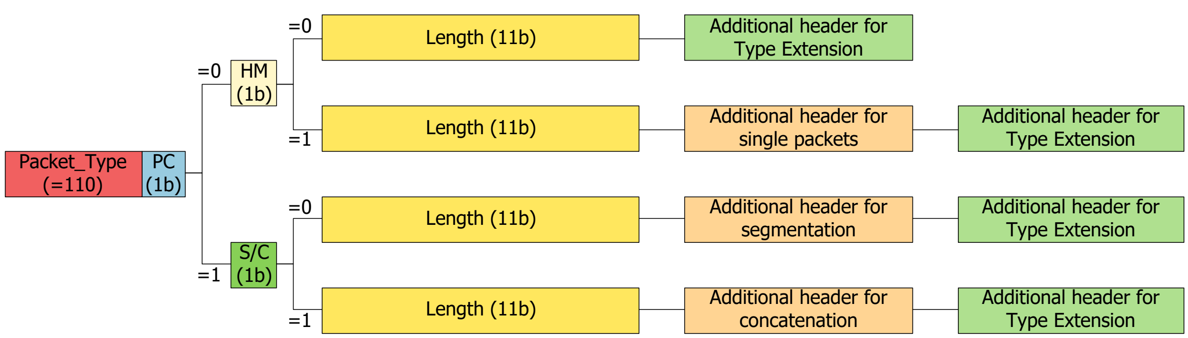

Figure 5.7 Structure of packet type extension (Base Header and Additional Header).

5.3.1 Additional Header for Type Extension

The Additional Header for Type Extension type_extension_hdr() shall consist of the fields as defined in Table 5.15. The text following Table 5.15 describes the semantics of each field in the table.

Table 5.15 Syntax of Additional Header for Type Extension

Syntax	No. of bits	Format
type_extension_hdr() {		
extended_type	16	uimsbf
}		

The semantic definitions for the fields are given below.

extended_type – This 16-bit field shall indicate the protocol or packet type of the input encapsulated in the ALP packet as payload. This field shall not be used for any protocol or packet type already defined in Table 5.2. In the current version of the specification, all extended_type values are reserved, as shown in Table 5.16.

Table 5.16 Code Values for Extended Type

extended_type	Meaning
0x0000–0xFFFF	Reserved

5.4 MPEG-2 TS Packet Encapsulation

This section specifies the ALP packet format when the input consists of MPEG-2 TS packets, that is, when the packet_type field of the Base Header is equal to '111'. Multiple TS packets can be encapsulated within each ALP packet. The number of encapsulated TS packets is signaled via the NUMTS field.

ALP provides overhead reduction mechanisms for MPEG-2 TS to enhance the transmission efficiency. The sync byte (0x47) of each TS packet is always deleted. The option to delete Null Packets and similar TS headers is also provided.

In order to avoid unnecessary transmission overhead, TS Null Packets (PID = 0x1FFF) may be removed. Deleted Null Packets can be recovered using the DNP field. The DNP field indicates the count of deleted Null Packets. The Null Packet deletion mechanism using the DNP field is described in Section 5.4.3.

In order to achieve more transmission efficiency, similar headers of consecutive MPEG-2 TS packets can be removed. When two or more successive TS packets have sequentially increasing continuity counter fields and other header fields of the TS packets are the same as each other, the header is sent once at the first packet and the subsequent headers are deleted. The HDM field shall indicate whether header deletion is performed or not. The detailed procedure of common TS header deletion is described in Section 5.4.4.

When multiple overhead reduction mechanisms are performed, overhead reduction shall be performed in the following sequence. First, sync removal, followed by Null Packet deletion, and then common header deletion.

5.4.1 ALP Packet Structure (for TS Packet Encapsulation)

The overall structure of the ALP packet header when using MPEG-2 TS packet encapsulation is depicted in Figure 5.8. The first byte is the Base Header and the optional second byte is the Additional Header, as shown in Figure 5.1. The Base Header length shall be one byte, and the minimum length of the ALP packet header in this case is one byte. When the Additional Header is present the total ALP packet header length is two bytes. Further details and the specific syntax are described in Table 5.17.

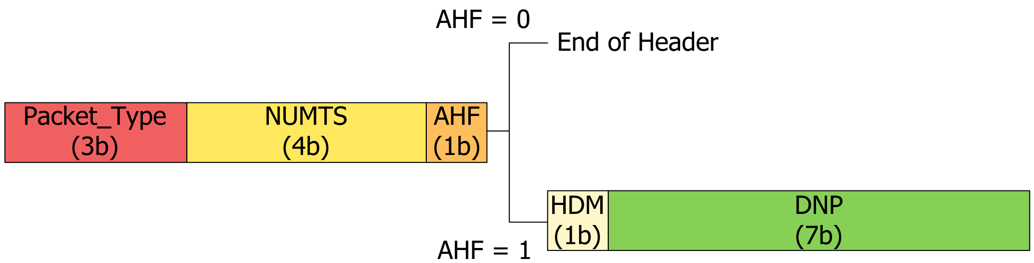

Figure 5.8 Structure of the header for MPEG-2 TS packet encapsulation.

The syntax of MPEG-2 TS packet encapsulation shall be as shown in Table 5.17. The text following Table 5.17 describes the semantics of each field in the table section.

Table 5.17 ATSC 3.0 Link Layer Packet Header Syntax for MPEG-2 TS

Syntax	No. of bits	Format
ATSC3.0_link_layer_packet() {		
packet_type	3	'111'
NUMTS	4	uimsbf
AHF	1	bslbf
if (AHF == 1) {		
HDM	1	bslbf
DNP	7	uimsbf
}		
}		

packet_type – This 3-bit field shall indicate the protocol type of the input packet as defined in Table 5.2. For MPEG-2 TS packet encapsulation, this field shall always be set to '111'.

NUMTS (Number of TS packets) – This 4-bit field shall indicate the number of TS packets in the payload of this ALP packet. A maximum of 16 TS packets can be supported in one ALP packet. The value of NUMTS = '0' shall indicate that 16 TS packets are carried by the payload of the ALP packet. For all other values of NUMTS, the indicated number of TS packets is recognized, e.g. NUMTS = '0001' means one TS packet is carried.

AHF (Additional Header Flag) – This field shall indicate whether the Additional Header is present or not. A value of '0' indicates that there is no Additional Header. A value of '1' indicates that an Additional Header of length 1-byte is present following the Base Header. When null TS packets are deleted or TS header compression is applied this field shall be set to '1'.

The Additional Header for TS packet encapsulation consists of the following two fields and is present only when the value of AHF in this ALP packet is set to '1'.

HDM (Header Deletion Mode) – This 1-bit field shall indicate whether TS header deletion is applied to this ALP packet. A value of '1' indicates that TS header deletion shall be applied as described in Section 5.4.4. A value of '0' indicates that the TS header deletion method is not applied to this ALP packet.

DNP (Deleted Null Packets) –This 7-bit field shall indicate the number of deleted null TS packets prior to this ALP packet. A maximum of 128 null TS packets can be deleted. When HDM = '0' the value of DNP = '0' shall indicate that 128 Null Packets are deleted. When HDM = '1' the value of DNP = ‘0’ shall indicate that no Null Packets are deleted. For all other values of DNP, the indicated number of Null Packets is recognized, e.g. DNP = ‘5’ means 5 Null Packets are deleted.

5.4.2 SYNC Byte Removal

When encapsulating TS packets into the payload of an ALP packet, the SYNC byte (0x47) from the start of each TS packet shall always be deleted. Hence the length of an MPEG-2 TS packet encapsulated in the payload of an ALP packet is always of length 187 bytes (instead of 188 bytes originally).

5.4.3 Null Packet Deletion

Transport Stream rules require that bit rates at the output of an emission multiplexer and at the input of a receiver’s de-multiplexer are constant in time and the end-to-end delay is also constant. For some Transport Stream input signals, Null Packets may be present in order to accommodate variable bitrate services in a constant bitrate stream. In this case, in order to avoid unnecessary transmission overhead, TS Null Packets (that is TS packets with PID = 0x1FFF) may be removed. The process is carried out in a way that the removed Null Packets can be re-inserted in the receiver in the exact place where they were originally, thus guaranteeing constant bitrate and avoiding the need for PCR (Program Clock Reference) time stamp updating.

Before generation of an ALP packet, a counter used for counting deleted Null Packets shall first be reset to zero and then incremented for each deleted Null Packet preceding the first useful packet to be encapsulated into the payload of the current ALP packet. The value of this counter is then used to set the DNP field. Then a group of consecutive useful TS packets is encapsulated into the payload of the current ALP packet and the value of each field in the ALP Packet’s header can be determined. After the generated ALP packet is injected to the physical layer, the DNP counter is reset to zero. When the DNP counter reaches its maximum allowed value, if the next packet is also a Null Packet, this Null Packet is kept as a useful packet and encapsulated into the payload of the current ALP packet. Each ALP packet shall contain at least one useful TS packet in its payload.

Figure 5.9 shows an example when Null Packet deletion is used. In this example HDM = ‘0’ and AHF = ‘1’ for both ALP packets. In the first ALP packet one Null Packet is deleted before two useful TS packets are transmitted in the ALP packet. The next packet is a Null Packet, so the ALP packet is completed and the DNP counter is reset to zero. In the header for this ALP packet NUMTS = ‘2’ and DNP = ‘1’. In the second ALP packet two Null Packets are deleted before transmission of 4 TS packets in the ALP packet. In the header for this ALP packet NUMTS = ‘4’ and DNP = ‘2’.

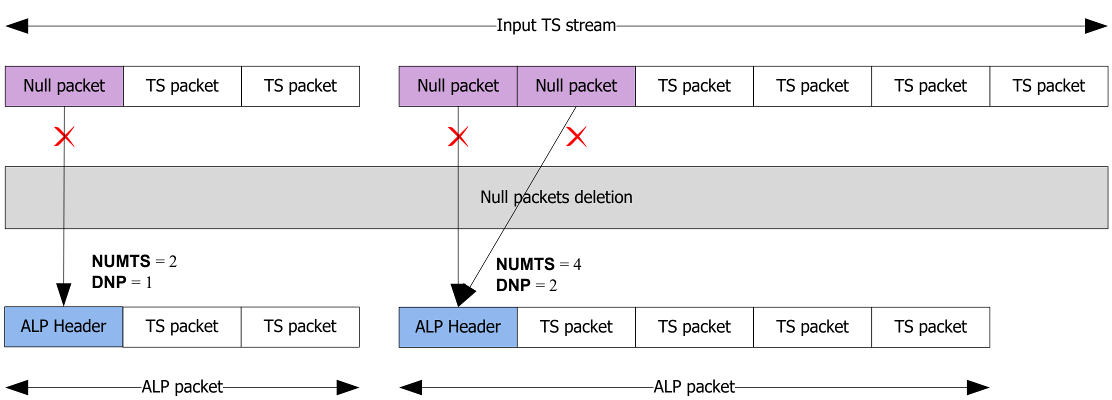

Figure 5.9 Null TS packet deletion example.

5.4.4 TS Packet Header Deletion

When two or more successive TS packets have sequentially increasing continuity counter fields whereas the other TS packet header fields remain the same, the TS packet header is sent once at the first TS packet and the other headers are deleted. When duplicate MPEG-2 TS packets [6] occur successively, the header deletion mechanism shall not be applied in the current ALP packet on the emission side. The HDM field shall indicate whether the header deletion is performed or not. When TS header deletion is performed, HDM shall be set to '1'.

Figure 5.10 illustrates the use of TS packet header deletion through an example. In this case four TS packets have the same header field values and the field NUMTS = '4'. HDM = '1' and DNP = '0' while AHF = '1'. In the receiver side, using the first packet header, the deleted packet headers are recovered, and the continuity counter is restored by incrementing it sequentially from that of the first header.

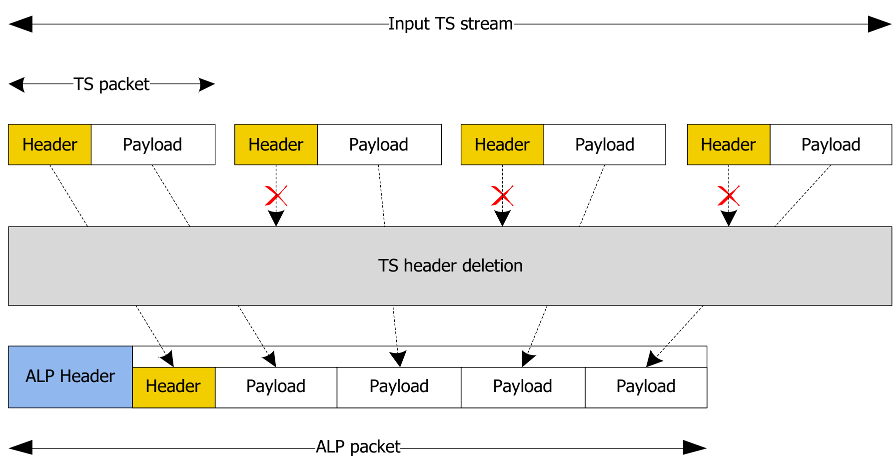

Figure 5.10 TS header deletion example.

6. IP HEADER COMPRESSION

At the ATSC 3.0 link layer, an IP header compression/decompression scheme is provided. The functional structure of IP header compression and decompression is shown in Figure 6.1.

IP header compression consists of the following three functions: pre-processing, header compression, and adaptation. The header compression scheme is based on the Robust Header Compression (ROHC) [7], [8]. In broadcasting applications, the pre-processing and adaptation functions are added.

On the emission side, the ROHC compression function reduces the size of the header for each IP packet. Next, an adaptation function extracts context information (i.e., header information that appears repeatedly in headers in the stream and that can be sent periodically and reinserted where needed) and builds signaling information from each packet stream to be sent separately in parallel.

On the receiver side, the adaptation function parses the signaling information associated with the received packet stream and reattaches context information to the received packet stream. The ROHC decompression function reconstructs the original IP packet by recovering the packet header.

Section 6.1 provides the method to configure simplified input to ROHC module for broadcast IP stream. Section 6.2 specifies references to the relevant RFCs and some restrictions of ROHC for ATSC 3.0 use. The detailed methods of packet configuration and context transmission are described in Section 6.3.

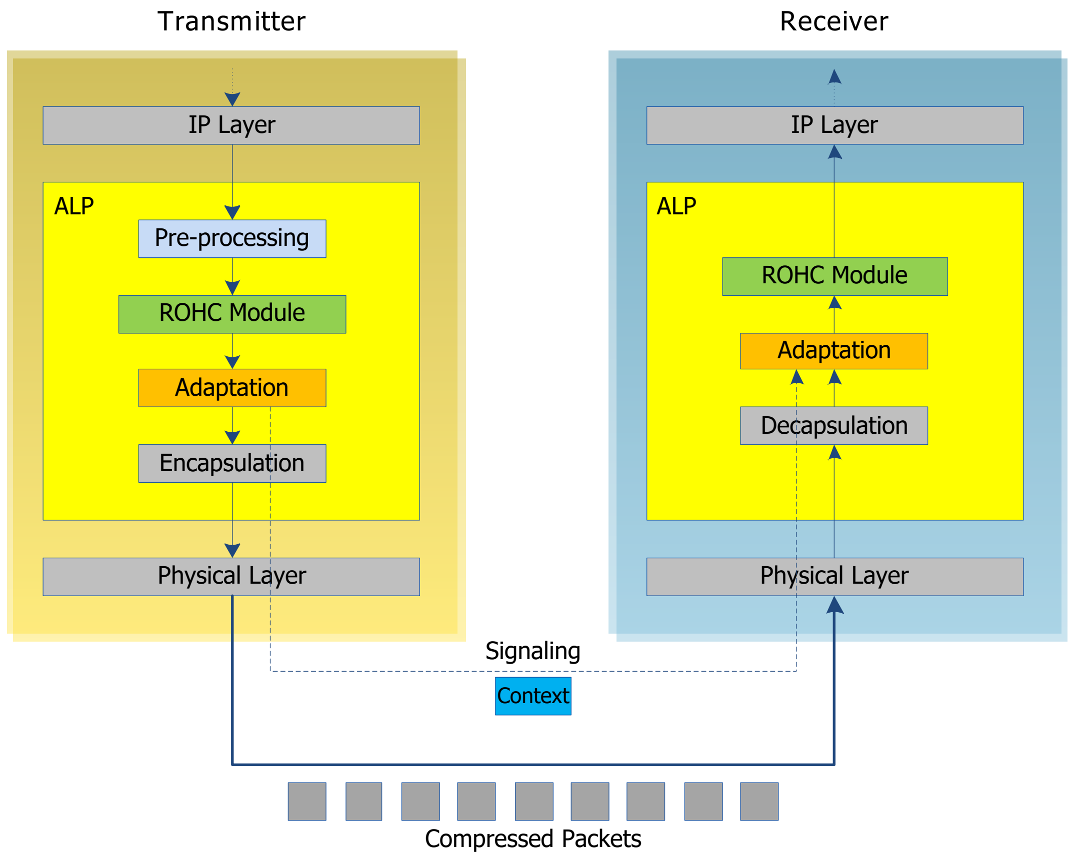

Figure 6.1 Functional structure of IP header compression and decompression.

6.1 Pre-processing for ROHC

In the emission side, for the optimal operation of IP header compression, the pre-processing module should be used. The purpose of the pre-processing module is to make an IP stream straightforward for the ROHC module. The pre-processing module does not create any impact on the operation of ROHC module and IP stack for the receiver side. It needs to be understood that the counterpart of the pre-processing module may not exist in the receiver.

When IP header compression is used, fragmented IP packets can cause additional complexity and result in inefficient ROHC operation. ROHC operation is simpler without IP fragmentation. Fragmented IP packets shall be re-assembled into the original unfragmented IP packets in the pre-processing module. When IP fragmentation is not present in the source stream, the pre-processing module is not needed.

6.2 ROHC-U (Robust Header Compression – Unidirectional Mode)

The IP header compression scheme shall be based on the Robust Header Compression. In consideration of characteristics of the ATSC 3.0 broadcasting link, the ROHC framework shall operate in the unidirectional mode (U-mode) as described in RFC 3095 [7], RFC 4815 [8] and RFC 5795 [9].

In the ROHC framework, multiple header compression profiles are defined. Each profile indicates a specific protocol combination, and the profile identifiers are allocated by the Internet Assigned Numbers Authority (IANA). For the ATSC 3.0 system, the '0x0002' profile shall be used. Table 6.1 shows the ROHC profiles defined for ATSC 3.0 system.

Table 6.1 ROHC profile for ATSC 3.0

Profile Identifier	Profile	Protocol Combination	Reference
0x0002	ROHC UDP	UDP/IP	RFC 3095 [7], RFC 4815 [8]

The ROHC framework defines channels to identify the compressed packet flow. In ATSC 3.0, the PLP should be mapped to an ROHC channel. Therefore, the CID should be managed separately at each PLP. In ROHC, each channel can define three types of CID configuration: small CID, 1-byte large CID, and 2-byte large CID. Among these configurations, small CID and/or 1-byte large CID should be configured for ATSC 3.0 systems.

For the protocol optimization, the following mechanisms and packet formats from the ROHC framework shall not be used.

- ROHC Padding – ALP does not provide a padding operation; however, the baseband packet of the ATSC 3.0 physical layer supports this operation. Thus, a padding operation shall not be performed in the ROHC compressor.
- ROHC Feedback – For ATSC 3.0, a unidirectional link is used. Therefore, the feedback mechanism and format shall not be used.
- ROHC Segmentation – The segmentation of ROHC shall not be used. If packet segmentation is required, it shall be performed in ALP encapsulation.

The ROHC framework defines configuration parameters to establish the context state between compressor and decompressor at each channel. In ATSC 3.0, the compressor sets these parameters and transmits signaling information to the decompressor.

MAX_CID – This ROHC framework parameter is the maximum CID value to be used by the compressor. This parameter is signaled by the max_CID field in the RDT (ROHC-U Description Table) as described in Section 7.1.2.

LARGE_CIDS – This Boolean ROHC framework parameter indicates the CID configuration. When this parameter is 'FALSE', the small CID is used in this channel. When this parameter is 'TRUE', the large CID is used in this channel. In ATSC 3.0, this parameter is inferred from the max_CID field in RDT. If max_CID is less than or equal to '15', LARGE_CIDS is considered as 'FALSE'. And if max_CID is larger than '15', LARGE_CIDS is considered as 'TRUE' in the decompressor.

PROFILES – This ROHC framework parameter defines which profile is used by the compressor. The list of profiles is defined in Table 6.1. This parameter is conveyed in the context_profile field of the RDT.

FEEDBACK_FOR – For ATSC 3.0, the feedback mechanism shall not be used. Therefore, this ROHC framework parameter shall not be used.

MRRU (Maximum Reconstructed Reception Unit) – In the ROHC framework, this parameter is defined for ROHC segmentation. Thus, this parameter shall not be used in ATSC 3.0.

According to the restriction of IP-ID field in A/331[2], it can be considered that there are two behaviors of IP-ID in the ROHC module; unused IP-ID and sequential IP-ID cases.

When IP fragmentation is not present, all IP-ID values of incoming IP packets are always ‘0x00’, and the value of the DF field is always ‘1’. In this case, IP-ID encoding shall not be used. The context value of IP-ID shall be set to ‘0x00’, and the context information of IP-ID is never updated. The context value of the DF in dynamic chain shall be set to ‘1’.

When IP fragmentation is either present or its presence is unknown, the IP-ID values of incoming IP packets increase monotonically, and the value of the DF field is always ‘0’. Even though the fragmented packets are reassembled prior to the ROHC module, the IP-ID fields shall be compressed. In this case, IP-ID shall be encoded with the offset IP-ID encoding scheme. The context of IP-ID shall be updated based on the offset IP-ID encoding scheme. The context value of the DF field in dynamic chain shall be set to ‘0’. Random IP-ID shall not be used.

The detailed procedures and algorithms of compression/decompression are specified in each RFC, and the implementation of the ROHC framework and compression algorithm is out of scope for this Standard.

6.3 Adaptation

In the case of transmission through a unidirectional link, if a receiver has no context information, the decompressor cannot recover the received packet header until it receives full context data. This may cause channel change delay and turn on delay. For this reason, context information and configuration parameters shall be sent in the RDT (see Section 7.1.2).

The Adaptation function at the transmitter provides construction of link layer signaling using configuration parameters and context information from the ROHC process. The adaptation function at the transmitter uses the previous configuration parameters and context information to transmit the link layer signaling periodically in each PHY frame. At the receiver, the fundamental process is reversed.

In addition, the Adaptation function provides out-of-band transmission of the configuration parameters and context information. Out-of-band transmission shall be achieved through link layer signaling. By these means, the Adaptation function reduces channel change delay and decompression error due to loss of context information.

6.3.1 Extraction of Context Information

6.3.1.1 Adaptation Mode 1

Figure 6.2 shows the procedure for context extraction when the emission side operates in adaptation mode 1. In adaptation mode 1, there is no additional operation performed on the original ROHC packet stream. The adaptation module shall operate as a buffer. Therefore, there is no context information such as static chain or dynamic chain in link layer signaling.

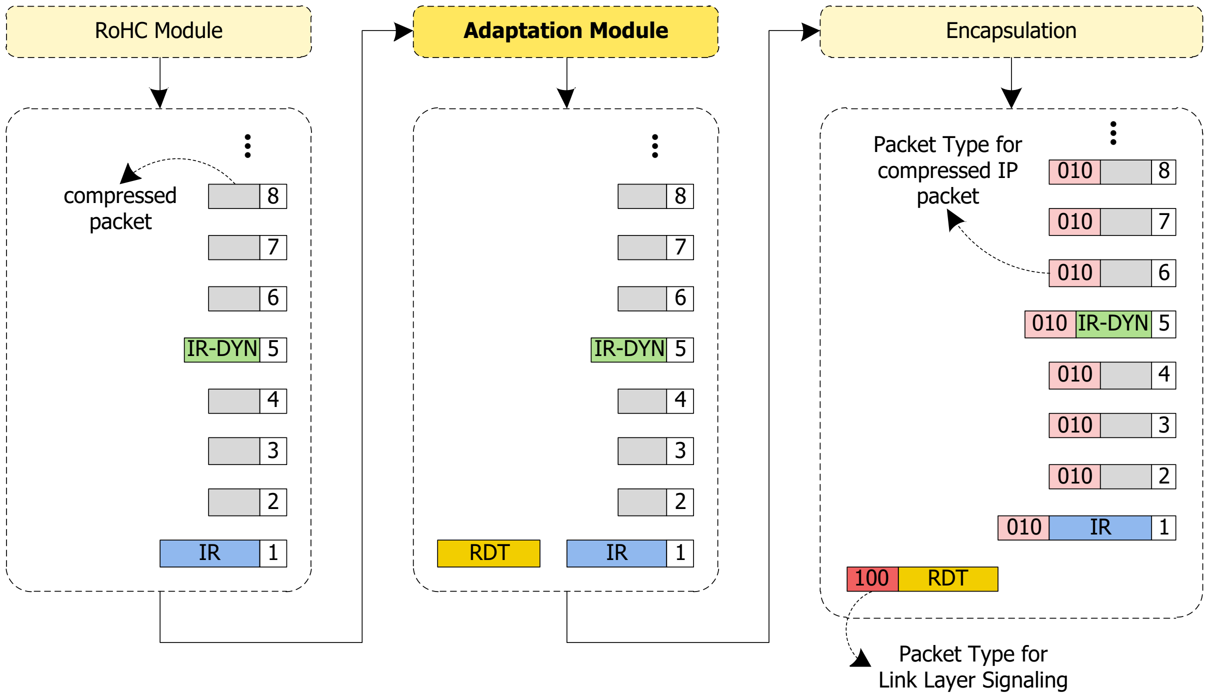

Figure 6.2 Procedure for IP header compression using adaptation mode 1.

6.3.1.2 Adaptation Mode 2

Figure 6.3 shows the procedure for context extraction when the emission side operates in adaptation mode 2. The adaptation module shall detect the IR packet from the ROHC packet flow and extract the context information (static chain). After extracting the context information, each IR packet shall be converted to an IR-DYN packet. The converted IR-DYN packet shall be included and transmitted inside the ROHC packet flow in the same order as the IR packet, replacing the original packet.

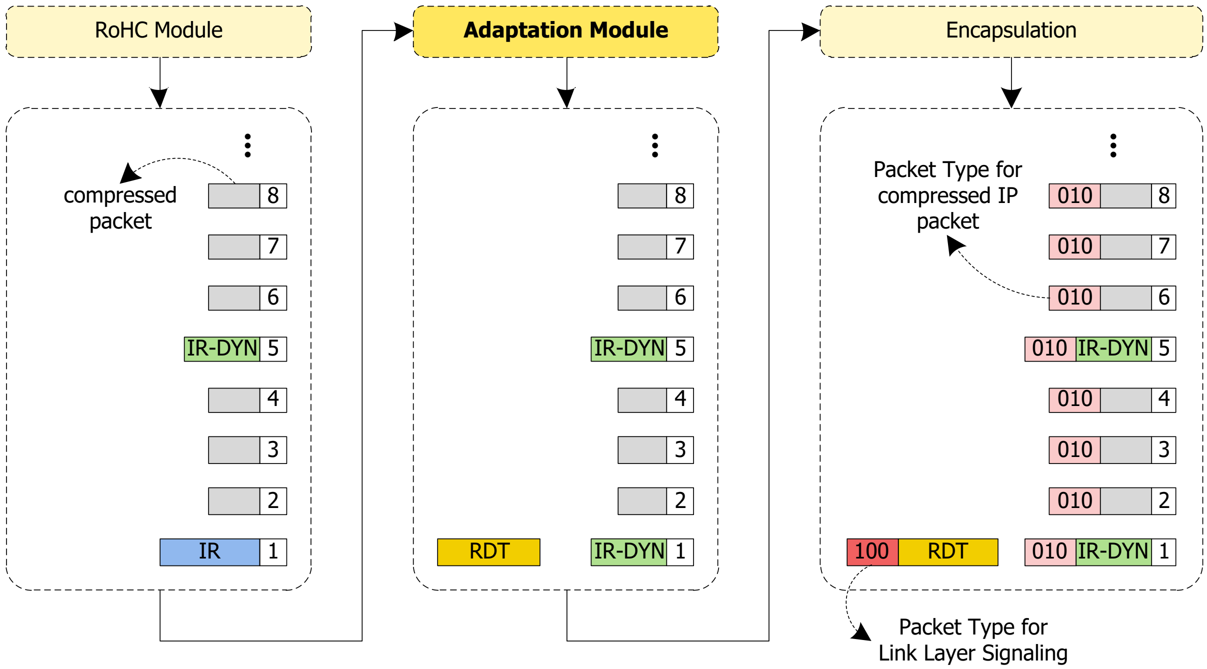

Figure 6.3 Procedure for IP header compression using adaptation mode 2.

Signaling (Context) information shall be encapsulated based on the transmission structure. Context information shall be configured as a part of the RDT and encapsulated within the link layer signaling as described in Section 5.2. In this case, the packet type value in the ALP packet carrying the link layer signaling shall be set to '100'.

6.3.1.3 Adaptation Mode 3

Figure 6.4 shows the procedure for context extraction when the emission side operates in adaptation mode 3. In adaptation mode 3, the adaptation module shall detect the IR and IR-DYN packets from the ROHC packet flow and shall extract the context information. The static chain and dynamic chain shall be extracted from the IR packet and the dynamic chain shall be extracted from the IR-DYN packet. After extracting the context information, each IR and IR-DYN packet shall be converted to a compressed packet. The converted compressed packet format shall be the same as the next compressed packet following the IR or IR-DYN packet. The converted compressed packet shall be included and transmitted inside the ROHC packet flow in the same order as the IR or IR-DYN packet, replacing the original packet.

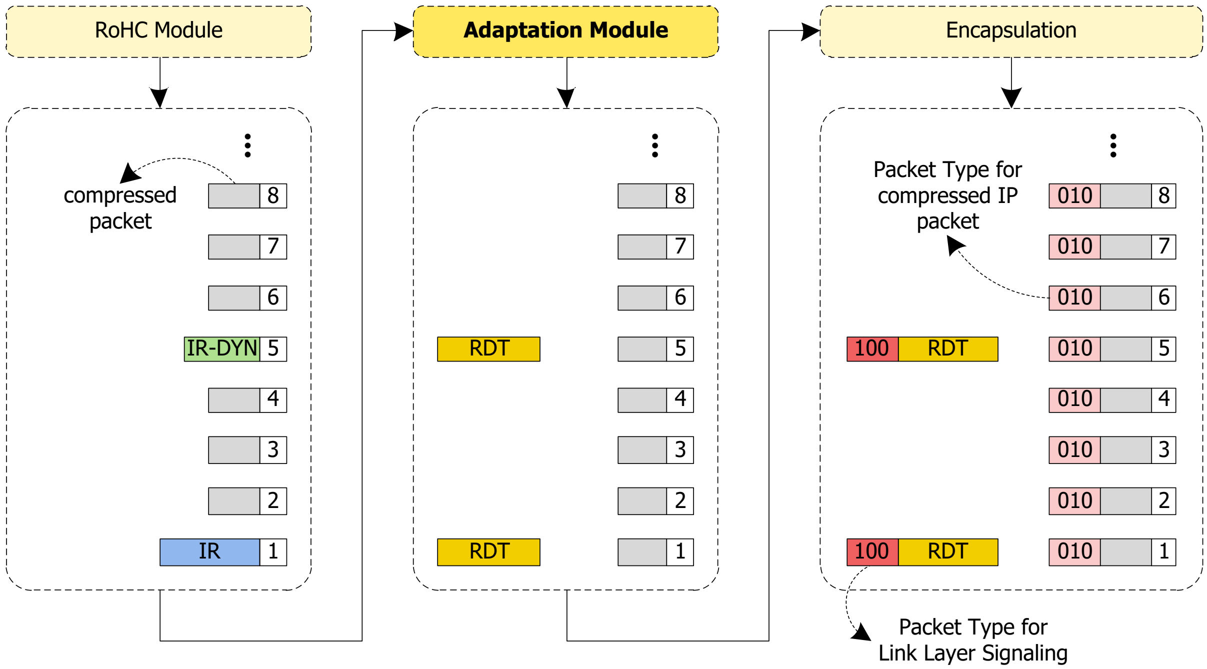

Figure 6.4 Procedure for IP header compression using adaptation mode 3.

Signaling (Context) information shall be encapsulated based on the transmission structure. Context information shall be configured as a part of the RDT and encapsulated to the link layer signaling as described in Section 5.2. In this case, the packet type value in ALP packet carrying the link layer signaling shall be set to '100'.

6.3.2 Transmission of Context Information

Each ALP stream delivered for a Service shall have its applicable RDT delivered prior to Service start. Delay of a required RDT relative to its Service RAP will result in delay of the Service start.

Two types of RDT delivery are defined in this Standard: Catalog RDT and Discrete RDT. A Catalog RDT can contain the context data for one or more ROHC/ALP streams, and is carried in a robust signaling PLP. Thus, it can supply multiple RDTs for multiple ALP streams as illustrated in Figure 6.5 and Figure 6.6. A Discrete RDT is carried exclusively within the ALP stream that it supports. This approach is consistent with RFC 3095 (ROHC), wherein the context is always carried within the related IP stream.

A Catalog RDT, if required for a Real Time Service, always shall be delivered at or before the entry point(s) into (all) the required ALP stream(s) after a Real Time Service RAP. A Catalog RDT may contain context information for an inactive ALP stream.

Discrete RDT(s), if required for a Real Time Service, always shall be delivered in the required ALP stream(s) immediately after a Real Time Service RAP.

An ALP stream utilized for Non-Real Time delivery shall have an applicable RDT delivered at least once every 5 seconds.

Figure 6.5 shows the transmission method which sends context information (RDT) separately from a compressed packet stream. The RDT, including extracted context information, may be transmitted separately from the related ROHC packet flow, along with other signaling data. An updated RDT shall be transmitted when the contained context information changes. The updated RDT shall be sent prior to any ALP packets reflecting the change. The RDT should be transmitted in each physical layer frame that contains a related Real Time Service RAP. If there are PLP(s) dedicated to the delivery of signaling on a given instance of ATSC 3.0 physical layer as denoted by a unique BSID, the RDT(s) associated with the related ALP streams may be delivered jointly with other link layer signaling carried in the same BSID.

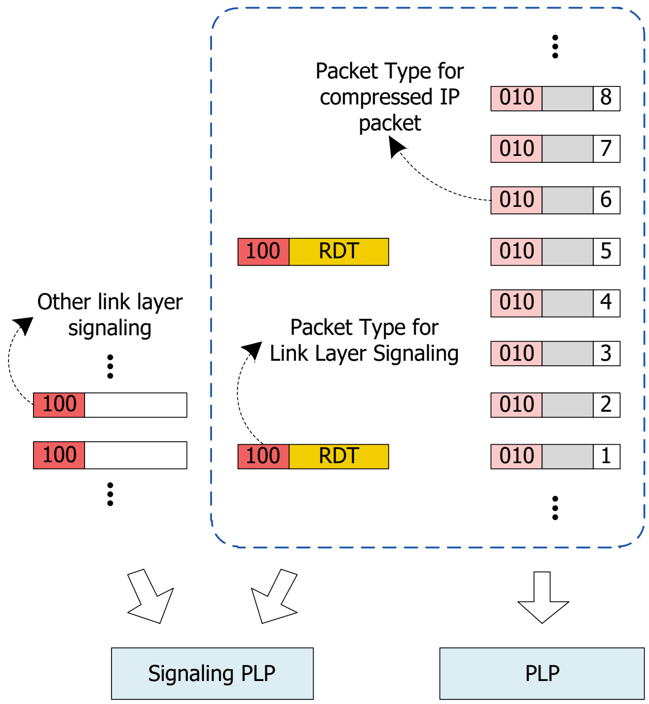

Figure 6.5 Transmission of context information.

The context information acquisition procedure in the receiver side is depicted in Figure 6.6. If there are PLP(s) dedicated to signaling in a given physical layer frame, the packets in the PLP(s) containing signaling shall be sent prior to all of the related ALP stream packets of that physical layer frame.

When the emission side operates in adaptation mode 1, the receiver must wait for the reception of an IR packet to start the decompression. Other packets which may be received prior to the first IR packet are not needed for decompression. The received ROHC packet flow is then sent to the ROHC decompressor.

When the emission side operates in adaptation mode 2, the receiver must wait for the reception of an IR-DYN packet to start the decompression. Other packets which may be received prior to the first IR-DYN packet are not needed for decompression. The adaptation module should parse the context information from the signaling data. Then, the adaptation module combines the context information (static chain) and the IR-DYN packets to form the ROHC packet flow. This operation is similar to receiving the IR packets. For the same context identifier, the IR-DYN packets are recovered to form the IR packets from the context information. The recovered ROHC packet flow is then sent to the ROHC decompressor.

When the emission side operates in adaptation mode 3, the receiver need not wait for any specific packet for decompression. In this case, the adaptation module combines the context information and the compressed packets to form the ROHC packet flow. This operation is similar to receiving the IR packets. For the same context identifier, the compressed packet is recovered to form the IR/IR-DYN packets from the context information. The recovered ROHC packet flow is then sent to the ROHC decompressor.

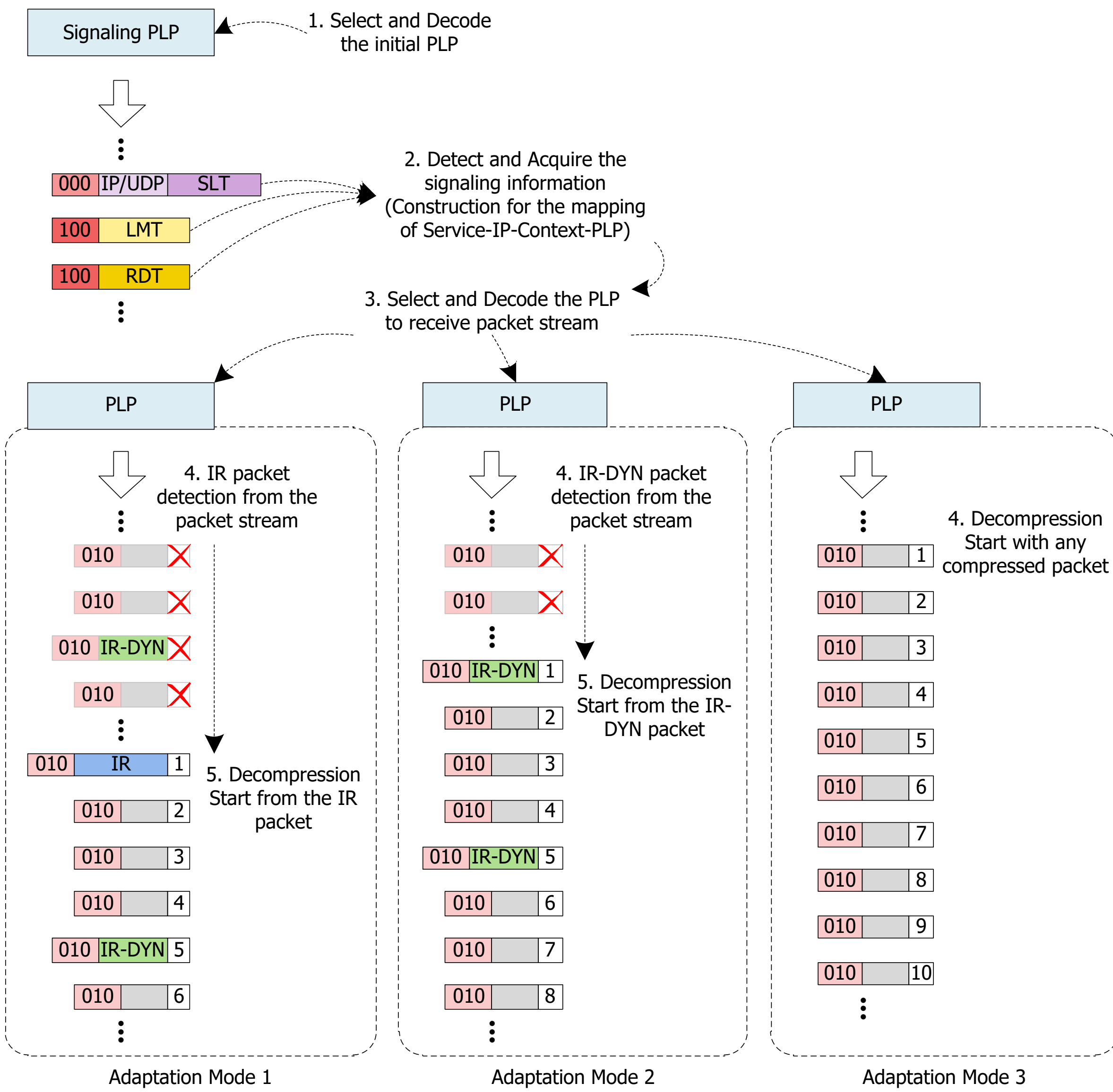

Figure 6.6 Context acquisition procedure in receiver side.

7. LINK LAYER SIGNALING

Generally, link layer signaling operates below the IP layer. In the receiver, the link layer signaling can be obtained earlier than the IP-level signaling such as Low Level Signaling (LLS), including the Service List Table (SLT) and Service Layer Signaling (SLS). Therefore, the link layer signaling can be obtained before upper-layer signaling is received.

The link layer signaling shall be encapsulated into ALP packets as described in Section 5.2. Each signaling table shall be the payload of an ALP packet. The link layer signaling can be transmitted in various formats, including binary and XML. All link layer signaling tables shall be transmitted in the same format.

7.1 Table Format for Link Layer Signaling

7.1.1 Link Mapping Table (LMT)

The Link Mapping Table (LMT) provides a list of multicasts carried in a PLP. The LMT also provides additional information for processing the ALP packets carrying the multicasts in the link layer. A link mapping example between upper layer and physical layer is shown in Figure 7.1. The LMT only provides information on UDP/IPv4 multicasts.

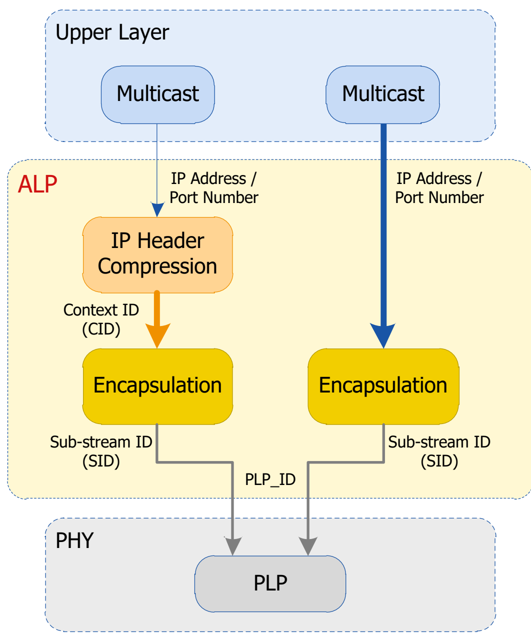

Figure 7.1 Example of Link Mapping for a PLP.

An LMT shall be present in any PLP identified as carrying LLS, as indicated by the value of the L1D_plp_lls_flag (see A/322 [1] Section 9.3.4) being set to '1'. Note that in the case where there are no multicasts referenced by LLS in a PLP, an LMT is still required to be present. Each instance of the LMT shall describe mappings between PLPs and IP addresses/ports for any IP address/port associated with any multicast referenced in the identified PLP carrying the LLS tables. However, the LMT shall not describe mappings for the multicast that is associated with LLS, namely multicasts with destination address 224.0.23.60 and destination port 4937/udp. Additionally, any LMT may describe mappings between PLPs and IP addresses/ports for any multicast, whether or not referenced in the identified PLP carrying the LLS tables.

The LMT, when present in a PLP, shall be repeated at a rate of at least once every 5 seconds. When present, the LMT should be broadcast at the same rate as the SLT.

The values for the Additional Header, which is defined in Table 5.11, shall be as given in Table 7.1. The syntax of the LMT shall be as given in Table 7.2.

Table 7.1 Additional Header Values for LMT

Fields in Additional Header for Signaling	No. of bits	Value
signaling_type	8	0x01
signaling_type_extension	16	0xFFFF
signaling_format	2	00
signaling_encoding	2	00

Table 7.2 Syntax for Link Mapping Table

Syntax	No. of bits	Format
link_mapping_table() {		
num_PLPs_minus1	6	uimsbf
reserved	2	‘11’
for (i=0; i<=num_PLPs_minus1; i++) {		
PLP_ID	6	uimsbf
reserved	2	‘11’
num_multicasts	8	uimsbf
for (j=0; j<num_multicasts; j++) {		
src_IP_add	32	uimsbf
dst_IP_add	32	uimsbf
src_UDP_port	16	uimsbf
dst_UDP_port	16	uimsbf
SID_flag	1	bslbf
compressed_flag	1	bslbf
reserved	6	‘111111’
if (SID_flag == 1) {		
SID	8	uimsbf
}		
if (compressed_flag == 1) {		
context_id	8	uimsbf
}		
}		
}		
}		

num_PLPs_minus1 – This 6-bit field shall have a value one less than the number of PLPs for which multicast-to-PLP mapping is provided in this table.

PLP_ID – This 6-bit field shall indicate the PLP corresponding to the multicast(s) signaled in this iteration of the “for” loop.

num_multicasts – This 8-bit unsigned integer field shall provide the number of multicasts carried in the PLP identified by the above PLP_ID field.

src_IP_add – This 32-bit unsigned integer field shall contain the source IPv4 address of the multicast signaled in this iteration of the “for” loop, carried in the PLP identified by the PLP_ID field.

dst_IP_add – This 32-bit unsigned integer field shall contain the destination IPv4 address of the multicast signaled in this iteration of the "for" loop, carried in the PLP identified by the PLP_ID field.

src_UDP_port – This 16-bit unsigned integer field shall represent the source UDP port number of the multicast signaled in this iteration of the "for" loop, carried in the PLP identified by the PLP_ID field.

dst_UDP_port – This 16-bit unsigned integer field shall represent the destination UDP port number of the multicast signaled in this iteration of the "for" loop, carried in the PLP identified by the PLP_ID field.

SID_flag – This 1-bit Boolean field shall indicate whether the ALP packet carrying the multicast identified by the above four fields, src_ip_add, dst_ip_add, src_udp_port and dst_udp_port, has an SID field in its optional header. When the value of this field is set to '0', the ALP packet carrying the multicast shall not have an SID field in its optional header. When the value of this field is set to '1', the ALP packet carrying the multicast shall have an SID field in its optional header and the value of the SID field shall be same as the following SID field in this table.

compressed_flag – This 1-bit Boolean field shall indicate whether header compression is applied to the ALP packets carrying the multicast identified by the four fields above, src_ip_add, dst_ip_add, src_udp_port and dst_udp_port. When the value of this field is set to '0', the ALP packet carrying the multicast shall have a value of '0x00' of packet_type field in its Base Header. When the value of this field is set to 1, the ALP packet carrying the multicast shall have a value of '0x02' in the packet_type field in its Base Header and the context_id field shall be present. When the value of compressed_flag is equal to 1, there shall be an RDT signaled that has PLP_ID value equal to the PLP_ID value in this table for this session.

SID – This 8-bit unsigned integer field shall indicate a sub-stream identifier for the ALP packets carrying the multicast identified by the above four fields, src_ip_add, dst_ip_add, src_udp_port and dst_udp_port. This field shall be present only when the value of SID_flag is equal to '1'.

context_id – This 8-bit unsigned integer field shall provide a reference for the context id (CID) provided in the ROHC-U description table as specified in Section 7.1.2 with the value of the PLP_ID field in the RDT equal to the value of PLP_ID in this table. This field shall be present only when the value of compressed_flag is equal to '1'.

7.1.2 ROHC-U Description Table (RDT)

As described in Section 6, the ROHC-U adaptation module generates the signaling including the static chain and some necessary information for header compression. The values for the Additional Header, which is defined in Table 5.11, shall be as given in Table 7.3. Table 7.4 describes the syntax of the RDT.

Table 7.3 Additional Header Values for RDT

Fields in Additional Header for Signaling	No. of bits	Value
signaling_type	8	0x02
signaling_type_extension	16	0xFFFF
signaling_format	2	00
signaling_encoding	2	00

Table 7.4 Syntax of ROHC-U Description Table

Syntax	No. of bits	Format
ROHC-U_description_table() {		
PLP_ID	6	uimsbf
max_CID	8	uimsbf
adaptation_mode	2	uimsbf
context_config	2	bslbf
reserved	6	'111111'
num_context	8	uimsbf
for (i=0; i<num_context; i++) {		
context_id	8	uimsbf
context_profile	8	uimsbf
if (context_config == 1) {		
context_length	8	uimsbf
static_chain_byte()	var	uimsbf
}		
else if (context_config == 2) {		
context_length	8	uimsbf
dynamic_chain_byte()	var	uimsbf
}		
else if (context_config == 3) {		
context_length	8	uimsbf
static_chain_byte()	var	uimsbf
dynamic_chain_byte()	var	uimsbf
}		
}		
}		

PLP_ID – This 6-bit field shall indicate the PLP corresponding to this table.

max_CID – This 8-bit field shall indicate the maximum value of context id to be used corresponding to this PLP. According to RFC 3095 [7], the maximum number of 1-byte large CID is 127. Therefore, the max_CID field shall not have a value greater than 127.

adaptation_mode – This 2-bit field shall indicate the mode of the adaptation module in this PLP as described in Section 6.3.1. The code values of adaptation_mode field shall be as given in Table 7.5.

Table 7.5 Code Values for Adaptation Mode

adaptation_mode	Meaning
00	Adaptation Mode 1 (see Section 6.3.1.1)
01	Adaptation Mode 2 (see Section 6.3.1.2)
10	Adaptation Mode 3 (see Section 6.3.1.3)
11	Reserved

context_config – This 2-bit field shall indicate the combination of the context information. If there is no context information in this table, this field shall be set to '0'. If static_chain_byte() or dynamic_chain_byte() is included in this table, this field shall be set to '1' or '2', respectively. If static_chain_byte() and dynamic_chain_byte() are both included in this table, this field shall be set to '3'.

num_context – This 8-bit field shall indicate the number of contexts in this table. The value of num_context shall not be larger than max_CID.

context_id – This 8-bit field shall indicate the context id (CID) of the compressed IP stream. In an ATSC 3.0 system, 8-bit CID shall be used for large CID. It shall be encoded as defined in Section 5.1.3 of RFC 3095 [7].

context_profile – This 8-bit field indicates the range of protocols used to compress the stream. It shall convey the eight least significant bits of the ROHC profile identifier defined in Section 6.1.

context_length – This 8-bit field indicates the length of the static_chain_byte(), dynamic_chain_byte() or both of static_chain_byte() and dynamic_chain_byte() sequence based on context_config.

static_chain_byte() – This field conveys the static information used to initialize the ROHC-U decompressor. The size and structure of this field depend on the context profile. The static_chain_byte() is defined in Section 5.7.7.1 of RFC 3095 [7] as "Static chain" in the sub-header information of the IR packet.

dynamic_chain_byte() – This field conveys the dynamic information used to initialize the ROHC-U decompressor. The size and structure of this field depend on the context profile. The dynamic_chain_byte() is defined in Section 5.7.7.1 of RFC 3095 [7] as sub-header information of IR packet and IR-DYN packet.

Annex A: ALP System Architecture

A.1 ALP SYSTEM ARCHITECTURE

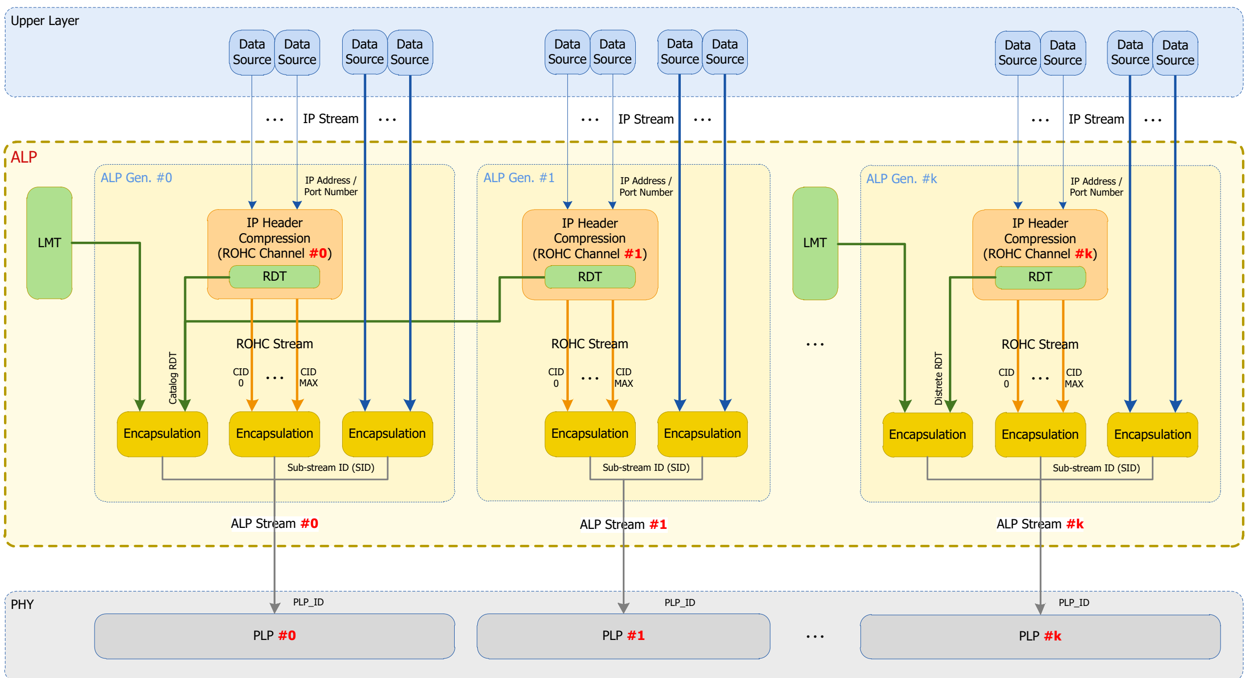

Figure A.1.1 ALP System Architecture and Internal identifiers

Figure A.1.1 shows the ALP system architecture and related identifiers in emission side.

One or more ALP streams can be delivered from an entire ALP generator. However, only a single ALP stream is delivered to each PLP. To generate ALP streams, a separate component ALP generator can be configured for each ALP stream. The ALP stream from each component ALP generator shall be delivered to its unique PLP. Therefore, the PLP ID can be used as ALP Stream ID and component ALP generator ID.

From Section 6.2, the ROHC framework defines channels to identify the compressed packet flow. In ATSC 3.0, single ROHC channel shall be configured in an ALP stream. Therefore, the PLP ID can be mapped to an ROHC channel number.

Annex B: ALP Packet Format Examples

B.1 ALP PACKET ENCAPSULATION

B.1.1 Single Packet Encapsulation

Figure B.1.1 shows an example of single short packet encapsulation.

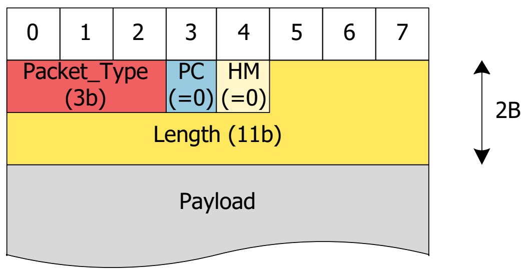

Figure B.1.1 Single Packet Encapsulation (short packet).

Figure B.1.2 shows an example of single long packet encapsulation. In this example, the SID and optional headers are not used.

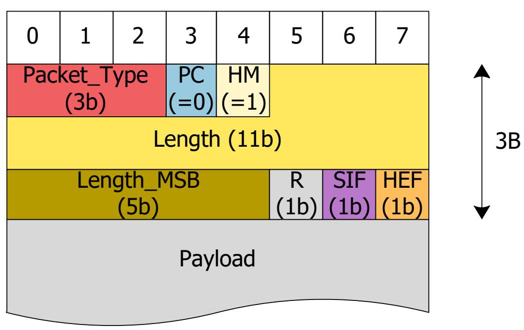

Figure B.1.2 Single Packet Encapsulation (long packet).

B.1.2 Segmentation

Figure B.1.3 shows an example where an IP packet is divided into segments, which are encapsulated in separate link layer packets.

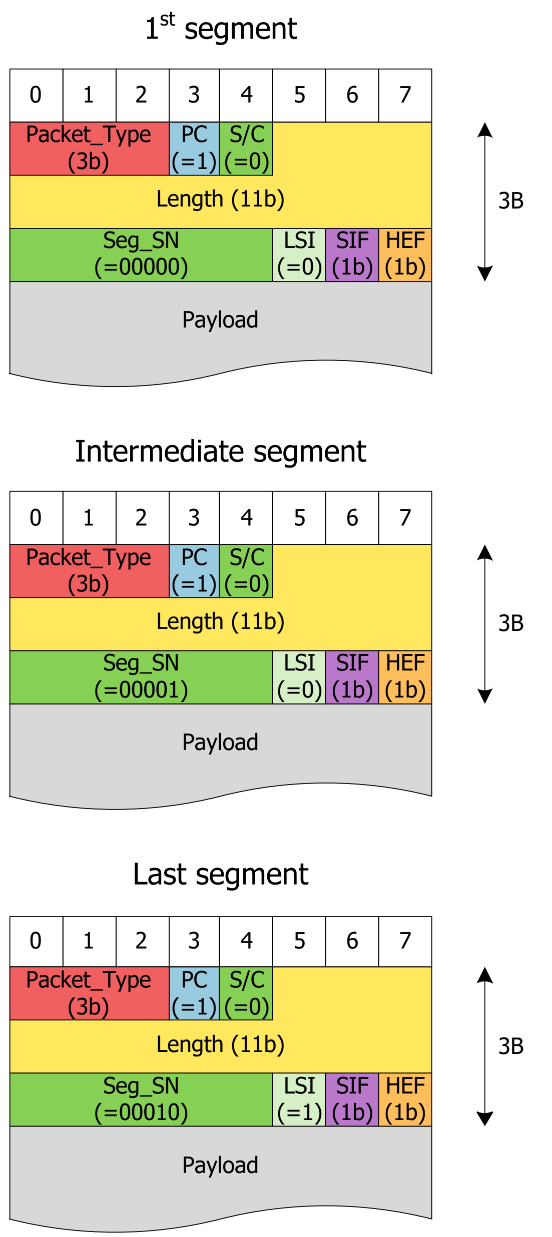

Figure B.1.3 Segmented Packet Encapsulation.

B.1.3 Concatenation

Figure B.1.4 shows an example where several IP packets are concatenated into the payload of one link layer packet. The component length fields (L_1 ~ L_n-1) indicate the length of each IP packet except the last one. The difference between the length field and the sum of all component length values indicates the length of the last concatenated IP packet.

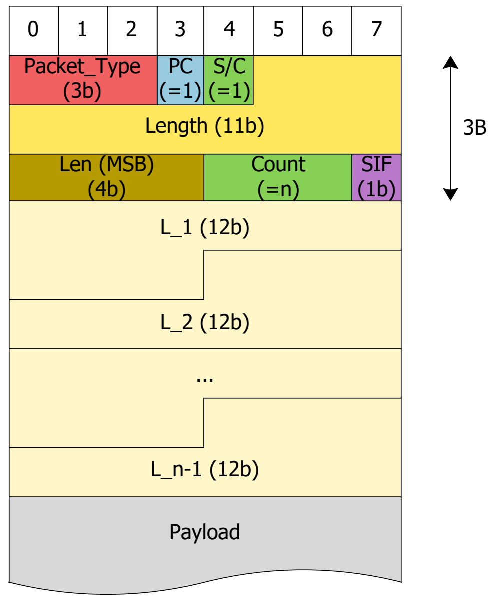

Figure B.1.4 Concatenated Packet Encapsulation (even number of component_length fields).

When an ALP header contains an odd number of component_length fields, four stuffing bits follow after the L_n-1 field as shown in Figure B.1.5.

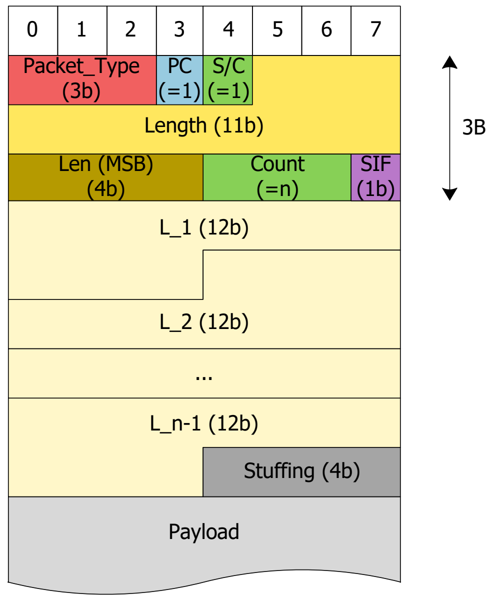

Figure B.1.5 Concatenated Packet Encapsulation (odd number of component_length fields).

B.2 MPEG-2 TS PACKET ENCAPSULATION

An ALP packet can carry several MPEG-2 TS packets without sync bytes in their payloads. Figure B.2.1 shows an example of an ALP packet containing eight MPEG-2 TS packets. The encapsulation process can be described as follows:

- Remove sync bytes for MPEG-2 TS packets to be encapsulated (the length of each MPEG-2 TS packet is reduced to 187 Bytes from 188 Bytes)
- Group eight MPEG-2 TS packets into the payload of an ALP packet (the total length of the payload is $187 \times 8 = 1496$ bytes)
- Generate an ALP header of length one byte with the following values:

  – packet_type = ‘111’

  – NUMTS = ‘1000’

  – AHF = ‘0’

The resulting ALP packet saves seven bytes compared with the case when the eight MPEG-TS packets are directly fed into the PHY layer.

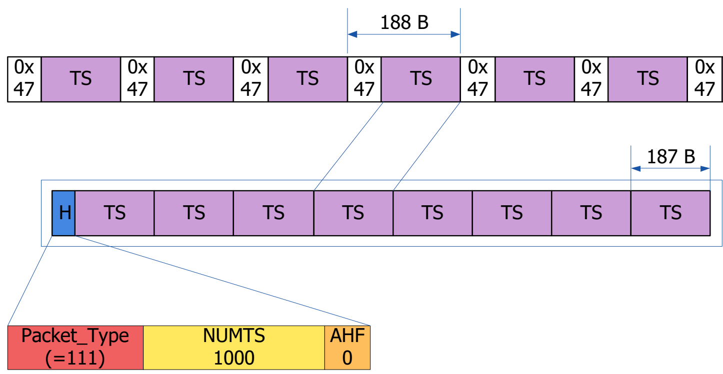

Figure B.2.1 MPEG-2 TS encapsulation example.

B.2.1 Using Null Packet Deletion

ALP can delete null MPEG-2 TS packets prior to the first MPEG-2 TS packet encapsulated into an ALP packet and let the receiver know the number of deleted null MPEG-2 TS packets via the header of the ALP packet. Figure B.2.2 shows an example of an ALP packet containing six MPEG-2 TS packets when two null MPEG-2 TS packets are deleted prior to the first MPEG-2 TS packet in the payload. The encapsulation process can be described as follows:

- Count and delete Null Packets
- Remove sync bytes for the MPEG-2 TS packets to be encapsulated (the length of each MPEG-2 TS packet is reduced to 187 Bytes from 188 Bytes)
- Group six MPEG-2 TS packets into the payload of an ALP packet (the total length of the payload is $187 \times 6 = 1122$ bytes)
- Generate an ALP header of length two bytes with the following values:
  - packet_type = ‘111’
  - NUMTS = ‘0110’
  - AHF = ‘1’ (It indicates that there are deleted Null Packets prior to the first MPEG-2 TS packet encapsulated into the payload.)
  - HDM = ‘0’
  - DNP = ‘0000010’

The resulting ALP packet is of length 1124 Bytes and saves 380 bytes compared with the case when the eight MPEG-TS packets are directly fed into the PHY layer.

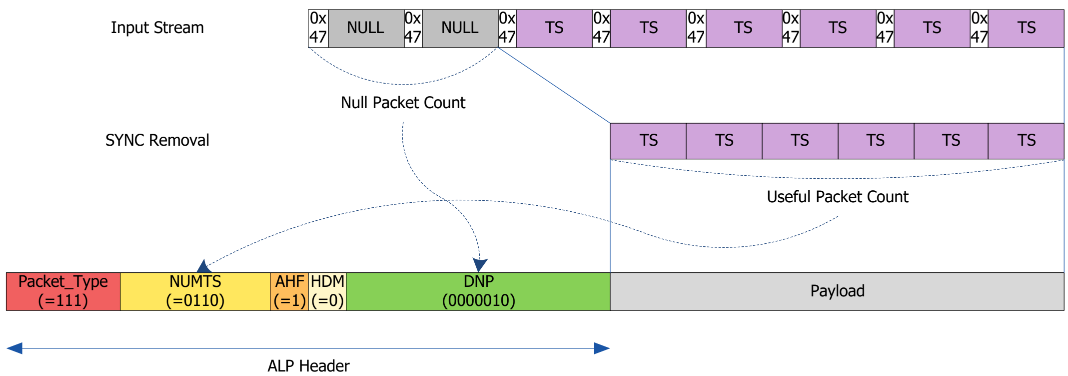

Figure B.2.2 MPEG-2 TS encapsulation example using Null Packet Deletion.

Figure B.2.3 shows an example of ALP packet decapsulation and Null Packet insertion. The decapsulation procedure at the receiver side can be described as follows.

- Check the DNP field.
- Get the number of TS packets in this ALP packet by using the NUMTS field.
- Insert the sync byte.
- Generate the appropriate number of Null Packets as indicated in DNP field before the group of useful TS packets.

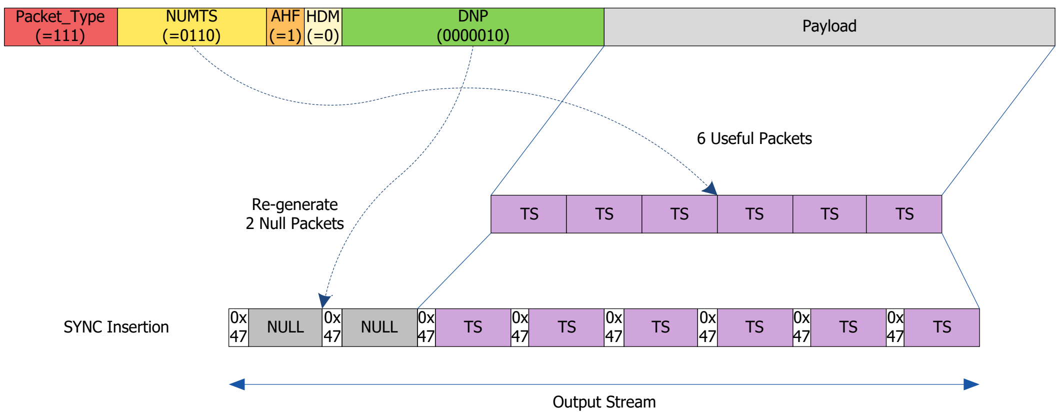

Figure B.2.3 MPEG-2 TS decapsulation example using Null Packet Deletion.

B.2.2 Using TS Header Deletion

ALP can further compress the headers of MPEG-2 TS packets encapsulated into an ALP packet. Figure B.2.4 shows an example of an ALP packet containing eight MPEG-2 TS packets which have the same header excluding the CC (continuity counter) field. The encapsulation process can be described as follows:

- Group eight TS packets which have the same header excluding the CC field.
- The header (excluding sync byte) is kept only for the first MPEG2-TS packet and is removed for the other seven MPEG-2 TS packets (the total length of payload is 3 + 184× 8 = 1475 Bytes).
- Generate an ALP header of length two bytes with the following values:

  – Packet_Type = ‘111’

  – NUMTS = ‘1000’

  – AHF = ‘1’

  – HDM = ‘1’

  – DNP = ‘0000000’

The resulting ALP packet is of length 1477 bytes and saves 27 bytes compared with the case when the eight MPEG-TS packets are directly fed into the PHY layer.

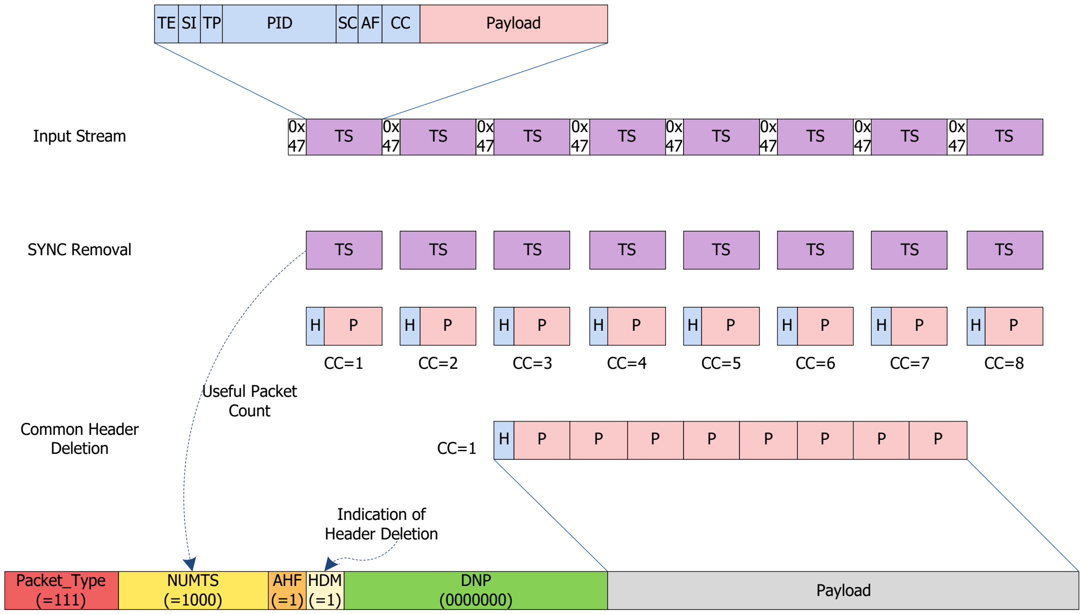

Figure B.2.4 MPEG-2 TS encapsulation example using TS header deletion.

Figure B.2.5 shows an example of ALP packet decapsulation and TS header recovery under the header deletion mode. The decapsulation procedure at the receiver side can be described as follows:

- Detect the header deletion mode by reading the HDM field
- Get the number of TS packets in this ALP packet by using the NUMTS field.
- The first TS packet includes a 3-byte header and 184-byte payload, and the remaining TS packets have a 184-byte payload only.
- Generate all TS packet headers using the header of the first TS packet, at this time each successive CC field is increased by one.
- Insert the sync byte.

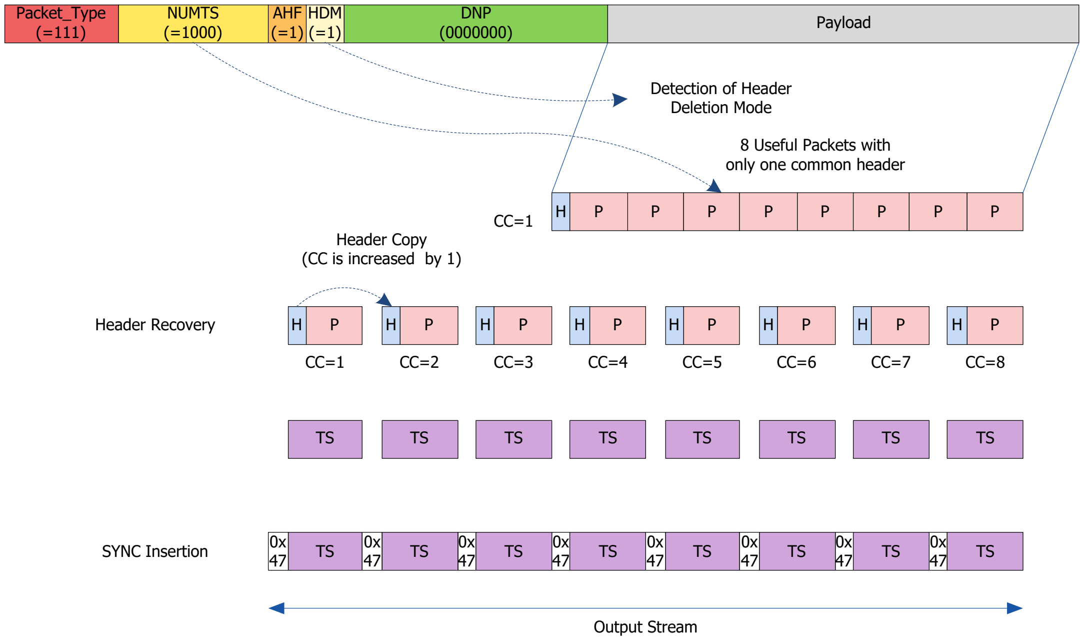

Figure B.2.5 MPEG-2 TS decapsulation example using TS header deletion.

***Annex C*: Compressed IP Packets**

This section describes the packet types for compressed IP packets as used in this Standard, when the ROHC profile is 0x0002.

C.1 INITIALIZATION OF CONTEXT

The ROHC module generates some flags in the context to identify the characteristic of each IP stream internally.

- **RND** – Flag indicating whether the IP-ID behaves randomly.
- **NBO** – Flag indicating whether the IP-ID is in Network Byte Order.

RND is one of the flags for configuration of ROHC packets. Some IPv4 stacks generate the IP Identifier values using a pseudo-random number generator. In the ROHC framework, the compressor detects such random behavior of the field. When the RND value is set to ‘1’, it means the IP Identifier is configured with a random number, and the IP identifier is sent as-is in its entirety as additional octets after the compressed header. According to the restriction of IP identifier in A/331[2] and Section 6.2, random IP Identifier is not used. Therefore, RND is always set to ‘0’. However, the procedure for the setting of these flags is out of scope for this document.

For ROHC UDP profile (0x0002), the compressor generates a 16-bit sequence number (SN) which increases by one for each packet received in the packet stream. The SN is added after the Checksum field for the dynamic part of UDP. This affects the format of dynamic chains in IR and IR-DYN packets.

C.1.1 Static Chain

This section describes the format of static chain. Figure C.1.1 describes the syntax of static chain which is defined in Sections 5.7.7.1, 5.7.7.4, 5.7.7.5 and 5.11.1 of RFC 3095 [7].

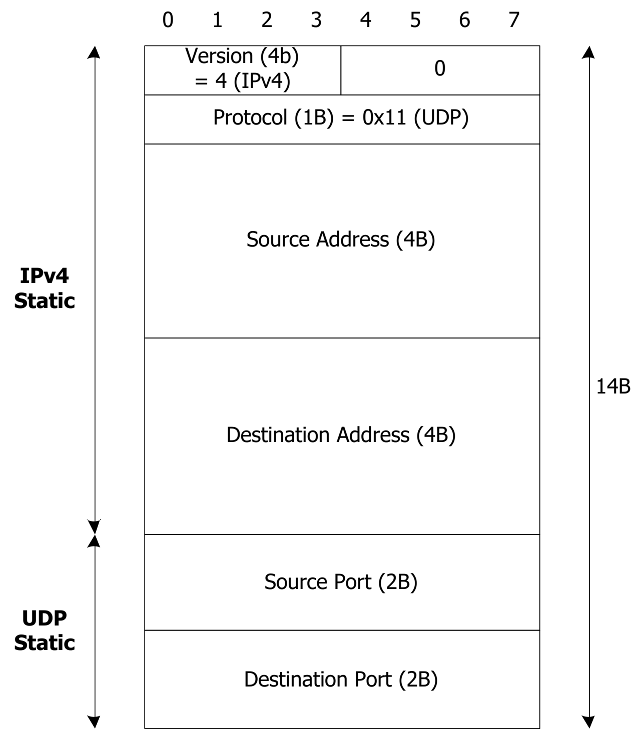

Figure C.1.1 Format of static chain

C.1.2 Dynamic Chain

This section describes the format of dynamic chain. Figure C.1.2 describes the syntax of dynamic chain which is defined in Sections 5.7.7.1, 5.7.7.4, 5.7.7.5 and 5.11.1 of RFC 3095 [7].

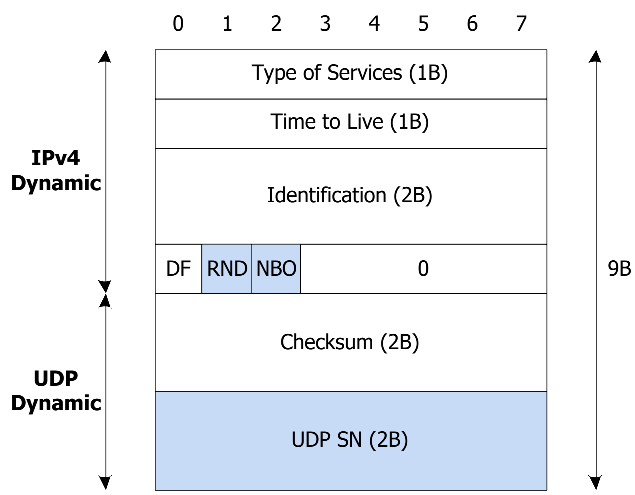

Figure C.1.2 Format of dynamic chain

The RND, NBO and UDP SN are the information generated in the ROHC compressor.

C.2 TYPES OF COMPRESSED IP PACKETS

The following packet types are used for IP/UDP header compression according to Section 5.11 of RFC 3095 [7]:

- IR
- IR-DYN
- UO-0
- UO-1
- UOR-2

Extension in UOR-2 should not be present in ATSC 3.0. As defined in RFC 3095 [7] Section 5.7.4, the flag X indicates whether the ROHC extension is present or not for the UOR-2 packet format. Therefore, the flag X shall be set to 0.

C.2.1 Small CID

C.2.1.1 IR and IR-DYN packet format

Figure C.2.1 and Figure C.2.2 indicate the general packet structures for IR and IR-DYN.

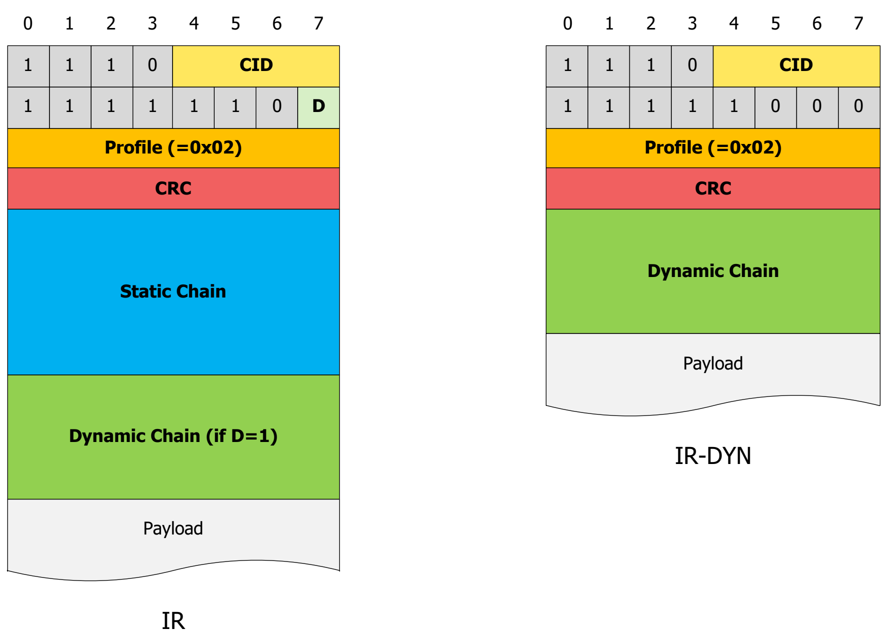

Figure C.2.1 IR and IR-DYN packet format (Small CID, CID ≠ 0)

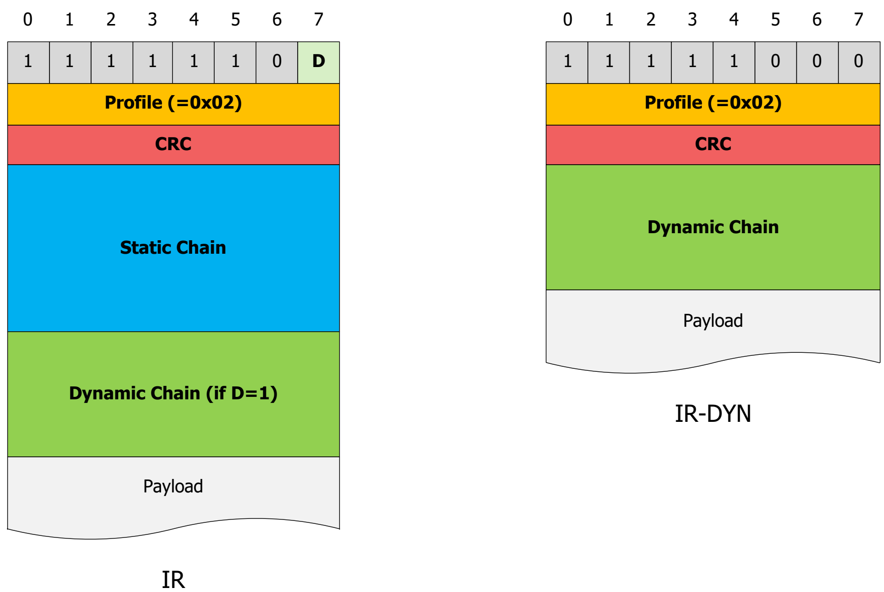

Figure C.2.2 IR and IR-DYN packet format (Small CID, CID = 0)

In the IR packet format, the flag `D` indicates whether the dynamic chain is present or not. When there is no dynamic chain in the IR packet header, the flag `D` is set to ‘0’. When there is a dynamic chain in the IR packet header, the flag `D` is set to ‘1’.

C.2.1.2 Compressed packet format

Figure C.2.3 and Figure C.2.4 indicate the packet structures for UO-0, UO-1 and UOR-2. In the UOR-2 packet format, the flag `X` shall be set to ‘0’, indicating the ROHC extension is not present.

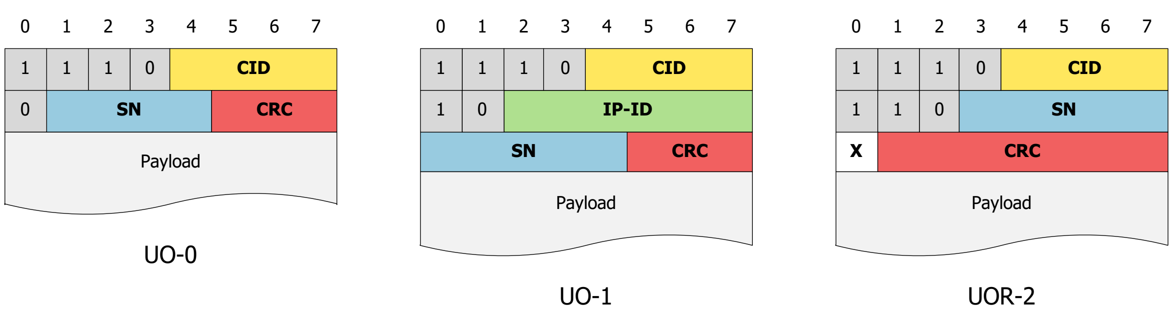

Figure C.2.3 Compressed IP packet format (Small CID, CID ≠ 0)

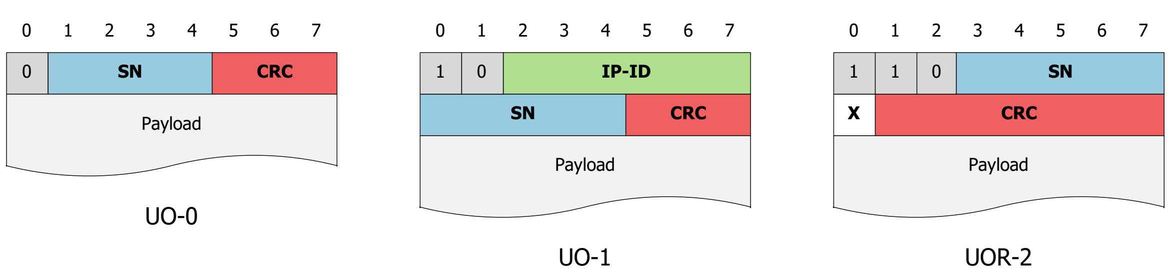

Figure C.2.4 Compressed IP packet format (Small CID, CID = 0)

C.2.2 Large CID

The ROHC framework defines two types of large CID: 1-byte large CID and 2-byte large CID. However, from these configurations, only the 1-byte large CID is configured for ATSC 3.0 systems.

C.2.2.1 IR and IR-DYN packet

Figure C.2.5 indicates the general packet structures for IR and IR-DYN, when 1-byte large CID is used.

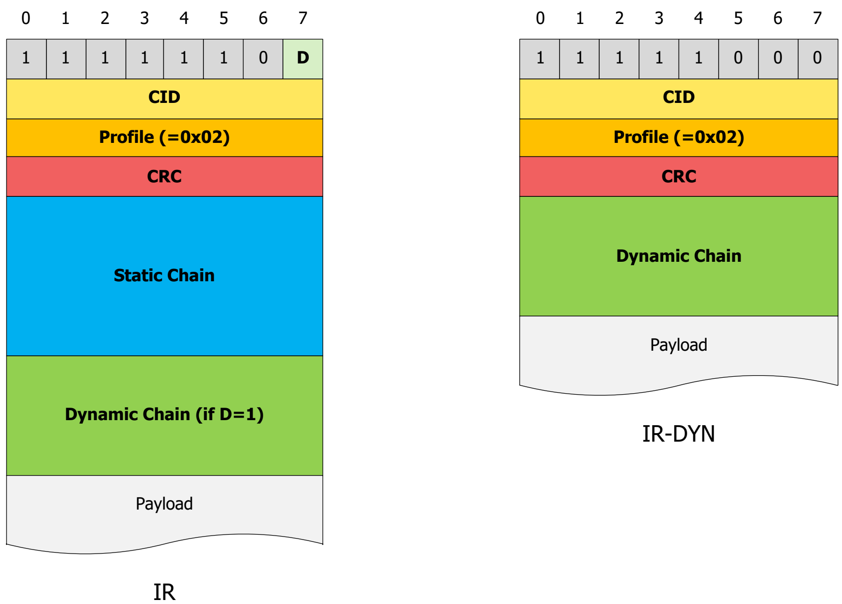

Figure C.2.5 IR and IR-DYN packet format (Large CID)

In the IR packet format, the flag D indicates whether the dynamic chain is present or not. When there is no dynamic chain in the IR packet header, the flag D is set to '0'. When there is a dynamic chain in the IR packet header, the flag D is set to '1'.

C.2.2.2 Compressed packet format

Figure C.2.6 indicates the packet structures for UO-0, UO-1 and UOR-2. In the UOR-2 packet format, the flag X shall be set to '0', indicating the ROHC extension is not present.

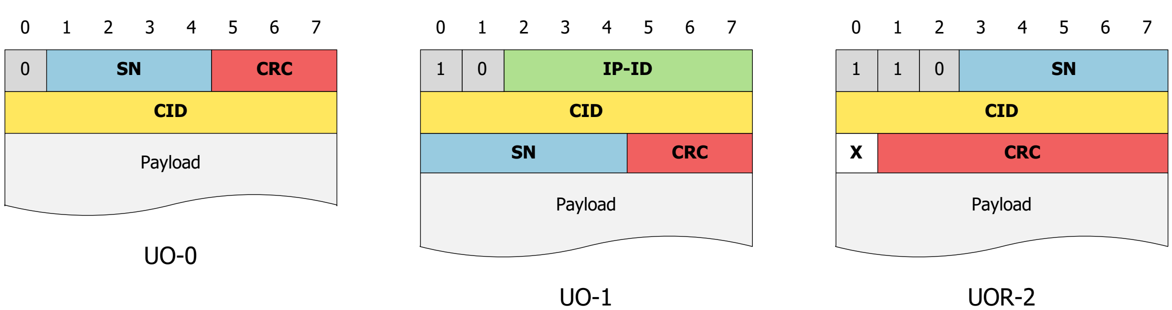

Figure C.2.6 Compressed IP packet format (Large CID)

Annex D: IP Header Compression Examples

D.1 USING ADAPTATION MODE 1

Figure D.1.1 shows an example of IP header compression when adaptation mode 1 is used.

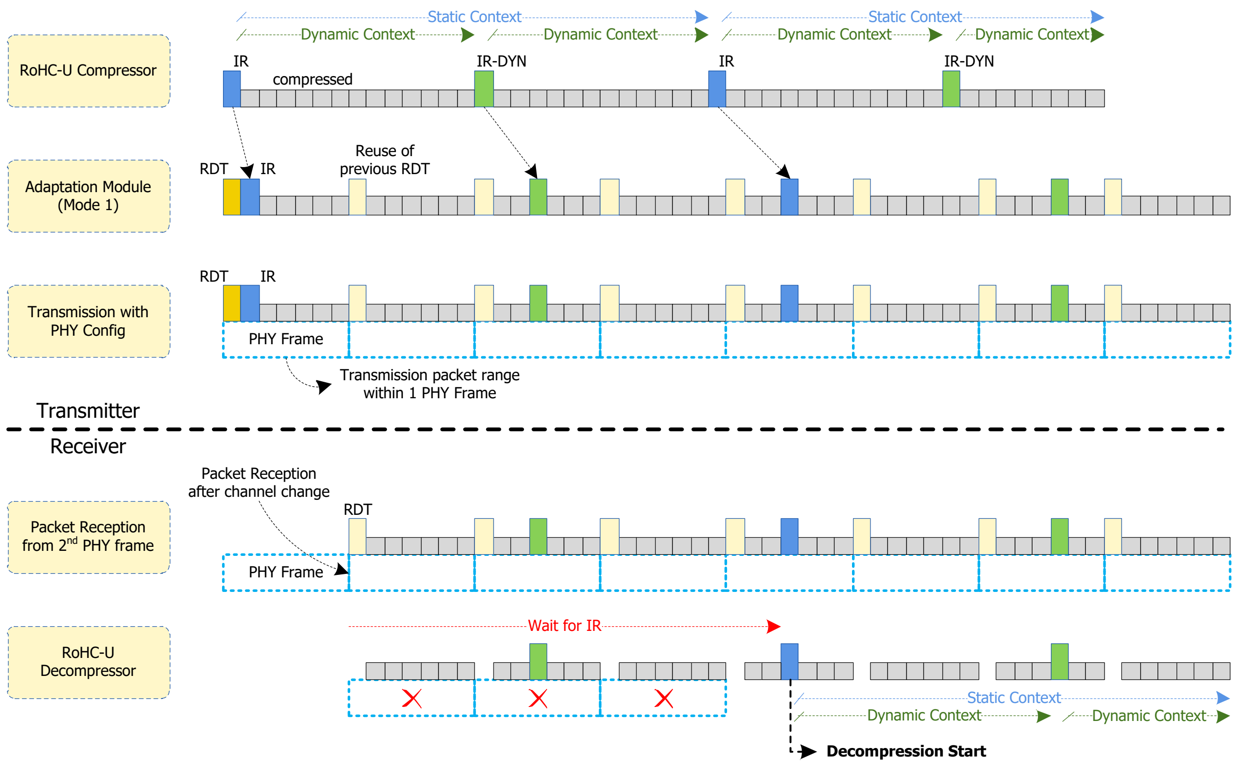

Figure D.1.1 Example of IP header compression using adaptation mode 1.

The operation in the emission side is described below.

When the IP packet stream is transported into the ROHC-U Compressor, for the first input IP packet, the context is initialized and the IR packet is generated. When the context is updated, the IR-DYN packet is generated. Static context is kept until the next IR packet is generated. The dynamic context is kept until the next IR or IR-DYN packet is generated.

When adaptation mode 1 is used, there are no procedures for packet conversion and context extraction. In order to transmit the ROHC configuration parameters, an RDT is generated in the adaptation module.

In this example, the RDT is transmitted every physical frame. If there is no change in configuration parameters, in order to transmit the RDT for every physical frame, the previous RDT can be reused. Packets are transmitted according to the capacity of the physical layer frame.

The operation in the receiver is described below.

In this example, it is considered that the packet stream is received from a second physical frame. From the first RDT, ROHC configuration parameters are obtained. In adaptation mode 1, the packet stream (not including the RDT) is passed into the ROHC-U decompressor. Since the ROHC-U decompressor cannot recover the IP header without full context, the ROHC-U decompressor must wait for the reception of the IR packet. After reception of the IR packet, the decompression process is started.

D.2 USING ADAPTATION MODE 2

Figure D.2.1 shows an example of IP header compression when adaptation mode 2 is used.

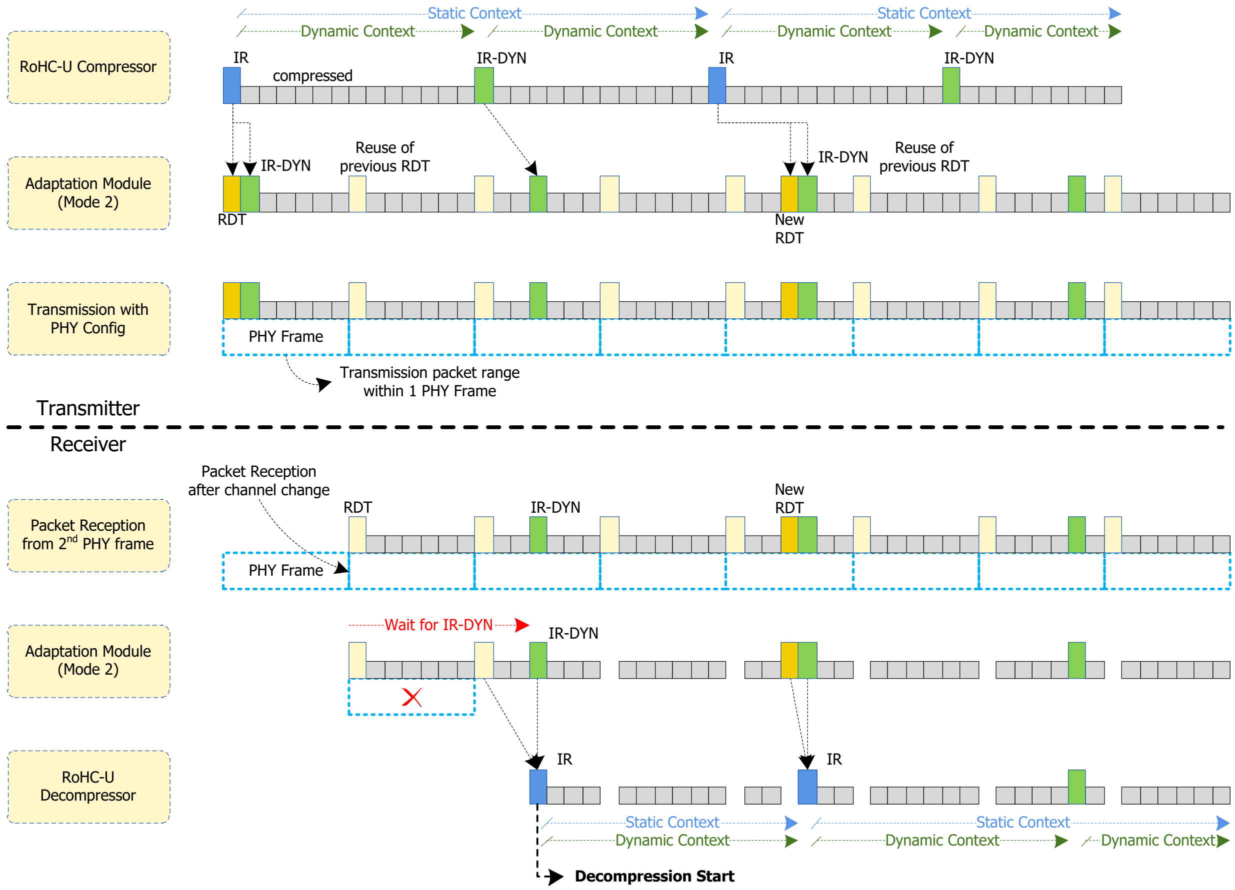

Figure D.2.1 Example of IP header compression using adaptation mode 2.

The operation in the emission side is described below.

When the IP packet stream is transported into the ROHC-U Compressor, for the first input IP packet, the context is initialized and the IR packet is generated. When the context is updated, the IR-DYN packet is generated. Static context is kept until the next IR packet is generated. The dynamic context is kept until the next IR or IR-DYN packet is generated.

When adaptation mode 2 is used, the adaptation module extracts the static context information from the IR packet. After extracting the context information, each IR packet is converted to an IR-DYN packet. Context information is configured as a part of the RDT with ROHC configuration parameters.

In this example, the RDT is transmitted every physical frame. If there is no change of static context information and configuration parameters, in order to transmit the RDT for every physical frame, the previous RDT can be reused. Packets are transmitted according to the capacity of the physical layer frame.

The operation in the receiver is described below.

In this example, it is considered that the packet stream is received from a second physical frame. From the first RDT, the static context information and ROHC configuration parameters can be obtained. In adaptation mode 2, the IR packet is recovered using the IR-DYN packet and the static context. Therefore, the adaptation module must wait for the reception of the IR-DYN packet. When the RDT is updated, the adaptation module recovers the IR packet.

The packet stream is passed into the ROHC-U decompressor after the IR packet recovery. The decompression process is started from the IR packet in the ROHC-U decompressor.

D.3 USING ADAPTATION MODE 3

Figure D.3.1 shows an example of IP header compression when adaptation mode 3 is used.

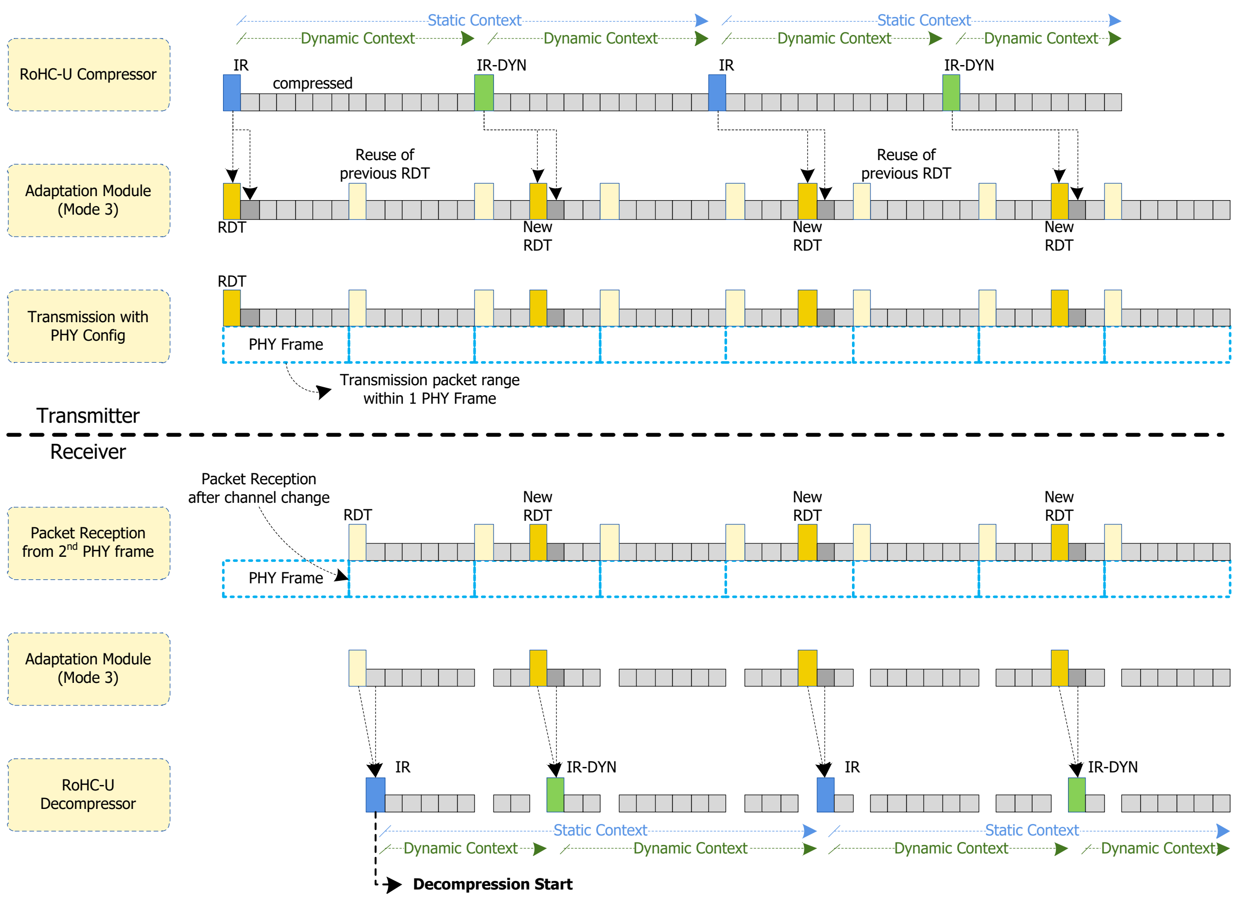

Figure D.3.1 Example of IP header compression using adaptation mode 3.

The operation in the emission side is described below.

When the IP packet stream is transported into the ROHC-U Compressor, for the first input IP packet, the context is initialized and the IR packet is generated. When the context is updated, the

IR-DYN packet is generated. Static context is kept until the next IR packet is generated. The dynamic context is kept until the next IR or IR-DYN packet is generated.

When adaptation mode 3 is used, the adaptation module extracts the static context and dynamic context information from the IR packet and the IR-DYN respectively. After extracting the context information, each IR and IR-DYN packet is converted to a compressed packet. Context information is configured as a part of the RDT with ROHC configuration parameters.

In this example, the RDT is transmitted every physical frame. If there is no change of context information and configuration parameters, in order to transmit the RDT for every physical frame, the previous RDT can be reused. Packets are transmitted according to the capacity of the physical layer frame.

The operation in the receiver is described below.

In this example, it is considered that the packet stream is received from a second physical frame. From the first RDT, the static/dynamic context information and ROHC configuration parameters can be obtained. In adaptation mode 3, the IR packet is recovered using any compressed packet with context information of the RDT. When the RDT is updated, the adaptation module recovers the IR or IR-DYN packet.

The packet stream is passed into the ROHC-U decompressor after IR packet recovery. The decompression process is started from the IR packet in the ROHC-U decompressor.

Annex E: ROHC Synchronization

E.1 SEQUENCE NUMBER GENERATION

Figure E.1.1 shows an example of sequence number (SN) generation and mapping in the emission ROHC-U compressor.

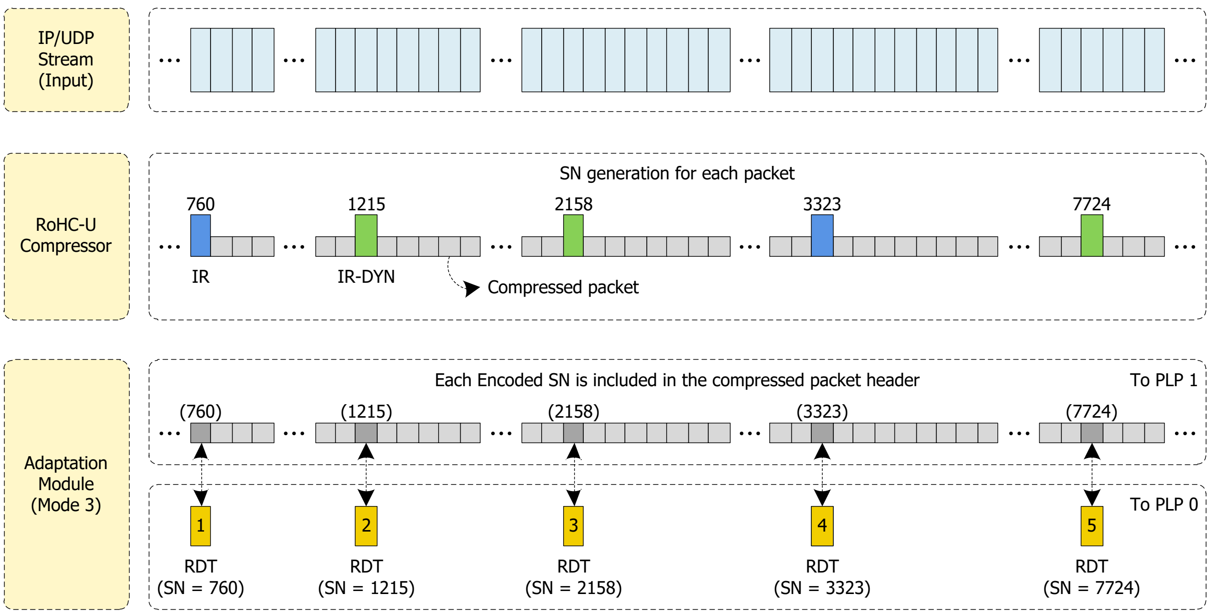

Figure E.1.1 Example of sequence number generation and mapping in the emission side.

According to RFC 3095 [7], for ROHC UDP profile (0x0002), the compressor generates a 16-bit sequence number (SN) which increases by one for each packet received in the packet stream. The SN is added after the Checksum field for the dynamic part of UDP as described in Figure C.1.2.

Each compressed packet header contains an encoded (compressed) SN and the length of the encoded SN can be 4 or 5 bits according to the packet format (UO-0, UO-1, and UOR-2). Each dynamic chain contains full-length (16-bit) SN.

In the emission, the main role of the adaptation function is the generation of the RDT. In order to generate the RDT, the adaptation module needs to gather the information from the ROHC compressor. This operation is described in Section 6.3. The RDT includes the context information of IR packets or IR-DYN packets. Therefore, the full-length SN is included in the RDT when the dynamic chain is contained.

E.2 SYNCHRONIZATION BETWEEN RDT AND COMPRESSED PACKET

When the RDT and compressed packet stream are transmitted through different PLPs, the context information is received separately from the related compressed packet. In this case, the synchronization between RDT and compressed packets has to be performed using the sequence number of dynamic chain and compressed packet header. The synchronization process is performed in the adaptation module in each receiver.

In the receiver, according to the emission operations, the adaptation module processes the information of RDT. The adaptation module gives ROHC configuration parameters to the decompressor for parsing the ROHC packet header (MAX_CID, LARGE_CIDS and so on). This kind of processing needs to be provided outside the protocol of ROHC as described in the RFCs. The adaptation module can obtain the context information from the RDT. The context information is sent to the decompressor. In adaptation mode 2 and 3, context information can be acquired from the RDT. The adaptation module performs synchronization process using the context information in the received RDT. Adaptation modes 1 and 2 do not need to synchronize between the RDT and the compressed packet stream. However, the adaptation module needs to be implemented to process the content of RDT when the synchronization is not necessary.

Figure E.2.1 shows an example of synchronization between the RDTs and compressed packet stream when adaptation mode 3 is used.

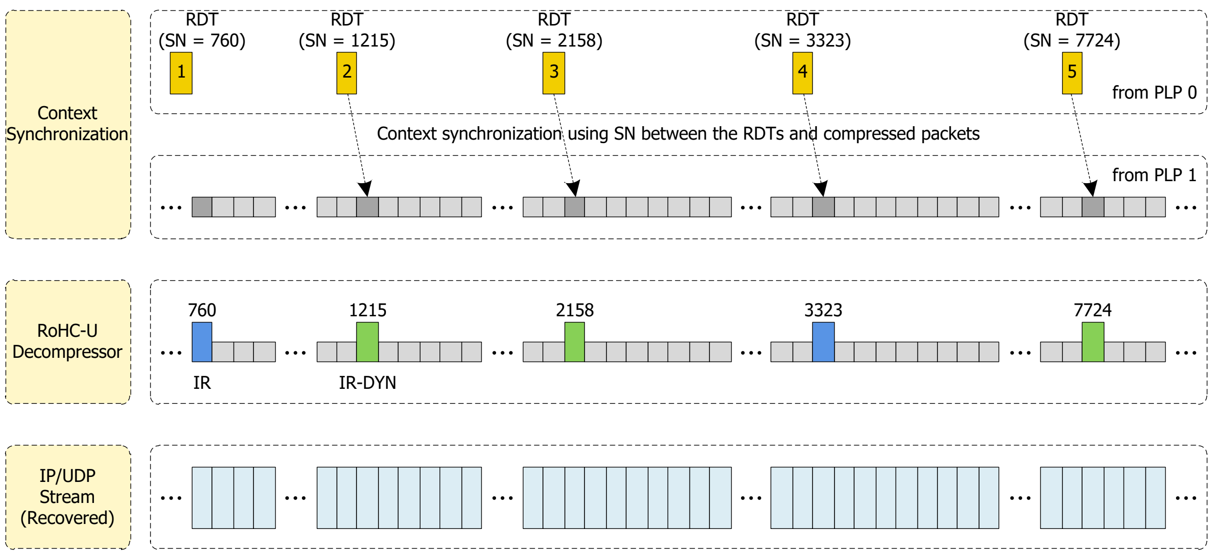

Figure E.2.1 Example of synchronization between the RDTs and compressed packets in the receiver.

The first step for synchronization is to find the reference SN for decompression. When the RDTs and compressed packet stream are received from the different PLPs, the adaptation module checks the SN from both of the RDT and compressed packets. In order to match the full-length SN from RDT and encoded SN in compressed packet, for example, the adaptation module may try the CRC check repeatedly until it passes. Other methods to find the reference SN can be also used based on the implementation structure of each receiver.

Hence, due to limited length of the compressed header, an encoded SN wraparound can occur. The repair of mismatch between the SN of RDT and encoded SN in each compressed header is specified in Sections 5.3.2.2.4 and 5.3.2.2.5 of RFC 3095 [7].

After the initial synchronization between the RDT and compressed packet stream, the input of the ROHC-U decompressor can be the same with the output ROHC packet stream of the ROHC-U compressor. According to the RFC, the SN for ROHC-U decompressor is always verified using

CRC and the verified SN is the updated reference SN. After the IP/UDP packet is recovered from the ROHC-U decompressor, the decompressed SN is used for the new reference SN, but this SN must not be sent to upper layers. After initial synchronization is completed, when the updated RDT is received, it is easy to match the SNs between the updated RDT and compressed packet due to the existence of the reference SN as described in Section 4.5.1 of RFC 3095 [7].

— End of Document —