SPECIFICATION OF THE DIGITAL AUDIO INTERFACE
(The AES/EBU interface)

Format for the user data channel

Tech. 3250–E – Supplement 1 – First edition August 1992

CONTENTS

European Broadcasting Union
Case Postale 67, CH–1218 Grand–Saconnex (Geneva) Switzerland

Preface

This document is a Supplement to the digital audio interface specification given in EBU document Tech. 3250, 2nd edition [1]. It was prepared as a result of a desire by users of the interface to have a recommended format for the user data channel provided by the user bit.

The requirement was for a system which is flexible, and independent of the user application, and is able to carry message data related in time to the audio data as well as information such as text which may be unrelated to the audio. A further requirement was for a constant data rate within a range of ± 12.5% of a sampling frequency of 48 kHz. The system specified is based on a widely–used packet communication protocol, High–level Data Link Control, HDLC [2] which is standardized in the information technology industry. The HDLC protocol has been adapted for unidirectional transmission and to permit the accurate transmission of time–dependent information; HDLC integrated circuits which are readily available from several manufacturers are still able to be used. It is therefore expected that this transmission format will be easy to implement without recourse to special hardware and, as a result, that it will be included as a matter of routine in commercial interface equipment.

This specification of the transport system is in two main parts. The first (*Section 4*) covers the *data formatting* and defines the mechanisms which transform the messages into packets, thereby allowing the multiplexing of simultaneous messages into the network. The second part (*Section 5*) describes the *channel management* and defines the use of the channel and the multiplexing rules.

The specification of the system characteristics follows closely the general principles of the ISO layer model.

The European Broadcasting Union is likely to agree on common uses and common formats for many regularly–used applications which will require the adoption of a common addressing strategy. It is therefore planned that addenda to the present specification will be published when agreements have been reached on specific topics of common interest.

Important notes

Conventions

To ensure coherence of presentation between EBU document Tech. 3250, Edition 2: *Specification of the digital audio interface (The AES/EBU interface)* and the present *Supplement* describing the user data channel format, the following conventions have been adopted. These conventions are based on the fact that the least–significant bit (LSB) of each data byte is transmitted first.

In *tables* showing the significance of each byte, nibble and bit, the binary codes read from left to right in the order LSB to MSB. Within each byte or nibble, codes are listed from top to bottom in increasing order of binary value. In bytes which contain two or more independent nibbles, the nibbles are listed from top to bottom in their order of transmission.

In *figures* showing the data–stream structure, the bytes are shown in order of transmission, reading from left to right. Within each byte, the bits are shown in order of transmission from left to right (consequently the LSB, which is always transmitted first in the AES/EBU interface, is on the left).

Differences between this specification and other published versions

The adoption of the conventions mentioned above results in some differences of presentation compared to other published versions of the specification of the user data channel format for the AES/EBU interface. No difference in technical specification is intended, except in respect of the following:

- *Message delimiters*, which were a feature of the draft specification of the user data channel format, have been abandoned in the present document, although they remain included in some published versions of this specification.
- *Section 4.2.1.b.* of this document includes additional clarification of the structure of the message length coding used when the message length is greater than 15 bytes and less than or equal to 4094 bytes.

Specification of the digital audio interface
(The AES/EBU interface)

Format for the user data channel

1. Scope

This document specifies a recommended method of formatting the user bit of each channel of the digital audio interface [1]. Each user data channel so formed is independent of the user application and is primarily intended for the transmission of data associated with the audio signal although data unrelated to the audio may also be transmitted. There is no restriction on the length of messages which may be transmitted in the user data channel. The format treats the user data channel as a transparent carrier with a constant data capacity for sampling frequencies in the range 42 kHz to 54 kHz i.e. 48 kHz ± 12.5%. There is no theoretical upper limit, but if sample rate conversion to a sampling frequency bgelow this range is performed, data management is to be employed to avoid loss of vital data.

The document describes the method of formatting user information into packets together with the rules for data insertion into the multiplex and for data management. A separate document is in preparation which recommends a strategy for hardware and software addressing. The purpose and content of the user information for particular applications are outside the scope of this document.

This document describes a practice for professional applications and is not intended to encompass applications related to consumer versions of the digital audio interface.

2. General

2.1. Messages

The AES/EBU digital audio interface is specified in [1] and is primarily intended to allow the transmission of audio signals between digital audio equipment. The present document specifies a system by which the user data channel can be used to transport a wide variety of messages along with the audio data. The messages can come from many different applications, scripts, sub–titles, editing information, copyright, performers credits, downstream switching instructions, etc. The application can be freely chosen and defined by the user and the messages which are sent can be of many different types.

2.2. Transport system

The transport system is based on a widely–used protocol, High–level Data Link Control, HDLC [2] which is standardized in the information technology industry. In general the HDLC system is capable of handling messages in two directions, but in this system it is only used in one, the same direction as the audio signals.

Integrated circuits are available from several manufactures which support the HDLC protocol. These often take the form of the HDLC interface combined with a microprocessor and a local interface to the message receiving or sending system.

2.3. Packets

Digital audio signals are likely to be repeatedly processed and passed on between different equipment in broadcasting studios. The messages which the user would like to accompany the audio signals are likely to grow at successive stages of the programme chain. This transport system allows downstream equipment to remove or add messages in the channel, provided there is enough capacity. Each message is sent as one or more packets, depending on its length. Each packet carries the address of its destination. This allows a receiver to read only those messages addressed to it. A large number of addresses is available and the system puts no restriction on the user as to how the addresses are used or for what purpose the messages are used.

2.4. Main features

The main features of the transport system from the point of view of the user are as follows:

2.4.1. Messages

The transport system can be used to carry a wide variety of information. Messages can be of any length, and they can be time critical or not.

2.4.2. Multiplexing

The transport system can carry simultaneous messages from different applications, subject to the maximum bit–rate provided by the interface. The system allows the user to insert messages at any point of the chain. The messages can have different priority levels, which affect the speed and frequency with which parts of a message can be inserted.

2.4.3. Bit rate

The transport system is based on packets and can be used with any audio sampling frequency. For audio sampling frequencies within the range of 48 kHz ± 12.5 % , which includes 44.1 kHz, it provides a constant message information rate. This is achieved by inserting dummy or justification bits at sampling frequencies above 42 kHz..

2.4.4. Synchronization

The transport system allows the user data channel to become a self–contained transmission channel which is independent of the audio signal block structure. However the channel can easily be synchronized to an external signal such as a time code clock or a video frame rate if this is required by the user.

2.4.5. Error detection

The transport system includes error checking which allows the detection of any corruption of the data in the messages. The user can safeguard important messages by instructing the transport system automatically to repeat every packet, or by repeating the message itself.

2.4.6. Efficiency

The system has an efficiency that is the ratio of the application data bit–rate to the total bit–rate of the user data channel.

The system needs a certain amount of overhead bit–rate to provide addressing, packet identification, error detection, justification etc. The percentage amount of this overhead varies predominantly with the length of the message and to a lesser extent with the block length.

The efficiency can be as high as 60% whith a block length of 40 ms and sampling frequency of 48 kHz. 70% efficiency is achievable with the same block length and a sampling frequency of 44.1 kHz because fewer justification bits are required.

3. Terminology

3.1. Transport system

The method by which messages are carried between the source and destination of the application.

3.2. Address

The identification of either the destination hardware or the application. It can be combined with an address extention.

3.3. Address extension

The part of the address which extends the range of destination hardware or applications that may be addressed.

3.4. Block

The repetitive structure chosen for the transport system by the user.

3.5. Continuity index

A count of the messages or packets with a given address. In this system the count is modulo–8, allowing up to seven missing messages or packets to be detected. This system uses a *message continuity index* and a *packet continuity index*.

3.6. Control byte

Contains information enabling receiving equipment to interpret and decode successfully and reliably the data which follow. The control byte has four fields for a *priority index*, a *packet continuity index*, *address extension enable* and *link bits*.

3.7. Cyclic redundancy check code (CRCC)

See *frame check sequence*.

3.8. Frame check sequence (FCS)

Two bytes formed mathematically from the packet information that are added to the packet to provide forward error detection.

3.9. High–level Data Link Control (HDLC)

An internationally–standardized protocol for the transmission of messages based on packets.

3.10. Frame level

The level of the transport system where the packets are formed into HDLC frames.

3.11. HDLC frame

The HDLC packet after it has been channel coded, with the addition of the frame check sequence and the limit flags [2].

3.12. HDLC frame flag

A unique code at the physical level which identifies the start and end of an HDLC frame.

3.13. Justification bits

The bits which are left unused in the user data channel to allow the system to operate between audio sample frequencies 42 kHz and 54 kHz without losing information. The number of justification bits varies with the audio sampling frequency.

3.14. A0 – least significant bit (LSB) of a byte

Bit transmitted *first* in this serial transmission system.

3.15. Message level

The level of the transport system where the messages are received from the application and processed so that they can be formed into packets. At the receiver, packets are processed, reformed into mesages and passed to the application.

3.16. Packet

A sequence of bytes comprising address and optional address extension bytes, control byte, and the message segment.

3.17. A7 – most significant bit (MSB) of a byte

Bit transmitted *last* in this serial transmission system.

3.18. Packet level

The level of the transport system where the self–contained packets that comprise the message are formed.

3.19. Physical level

The level of the transport system where the frames are channel coded and inserted in the user data channel of the digital audio interface.

3.20. Priority index

The number, chosen by the user, by which a priority is assigned to a packet or message. Four priority levels have been provided.

3.21. Repetition index

The number, chosen by the user, which controls the number of times each packet of a message is repeated.

3.21. Segment

Part of a message after it has been split up. The segment will form the information field of a packet.

4. Data formatting

4.1. The application layer

The messages are generated by the application, and are then passed to the transport system together with the other parameters which the transport system uses to process the messages. The parameters are:

– the destination address;

– a priority index;

– a repetition index.

4.1.1. Destination address

The application shall provide a destination address. It is either one byte long, or two bytes if an address extension is used.

4.1.2. Priority index

The priority index shall be chosen by the user according to the urgency of the message. Four priority levels are provided, and the choice determines either the maximum delay before the message is sent or how quickly it is sent if it is a long message. The priority system is described in more detail in *Section 5.3.2*.

4.1.3. Repetition index

The repetition index shall control how many times each packet of the message shall be repeated by the transport system. The repetition index can have any value, but to minimise the system loading a repetition index between 0 and 5 is recommended. The whole message can also be repeated by the application.

4.1.4. Message

The message content shall be treated as a binary signal and it can be of any number of 8–bit bytes.

4.2. The transport system

The transport system shall receive the message data from the application level and process it so that it can be transmitted in the user data channel. The data formatting shall be carried out on four levels (*Fig. 1*):

– message level;

– packet level;

– frame level;

– physical level.

Each level has its own function and all the levels together ensure the correct formatting of the data to allow it to be transported.

4.2.1. Message level

The message level (*Fig. 1* – step 1) shall receive the message from the application together with the other parameters listed in *Section 4.1*. It shall add a header to the message and then divide the message, including the header, into segments which will be transported separately. Finally the message level shall pass on the segments, together with the parameters, to the packet level.

a) Message header

The message header (*Fig. 2*) shall contain information on the message length and a message continuity index. The header is either one or two bytes long, depending on the length of the message. A message length format bit in the first byte shall signal whether the second byte, the message length extension, is used.

Message header – Byte 0		
Bits A0–A3		MSBs of message length code
		These bits indicate the length of the message; their precise use is described in *Section 4.2.1.b*).
Bit A4		Message length format
	0	Header is one byte long
	1	Header is two bytes long
Bits A5–A7		Message continuity index
		The three–bit continuity index shall be a modulo–8 count of the messages sent by an application. A7 is the MSB. The receiver can use this index to detect any loss of messages. Each application shall have its own continuity index. However when a message is repeated under the control of the repetition index the message continuity index shall not change (see *Section 4.2.2.b*).

Message header – Byte 1 (optional)	
Bits A0–A7	Message length extension code
	These bits allow the length code of the message to be extended as described in *Section 4.2.1.b*.

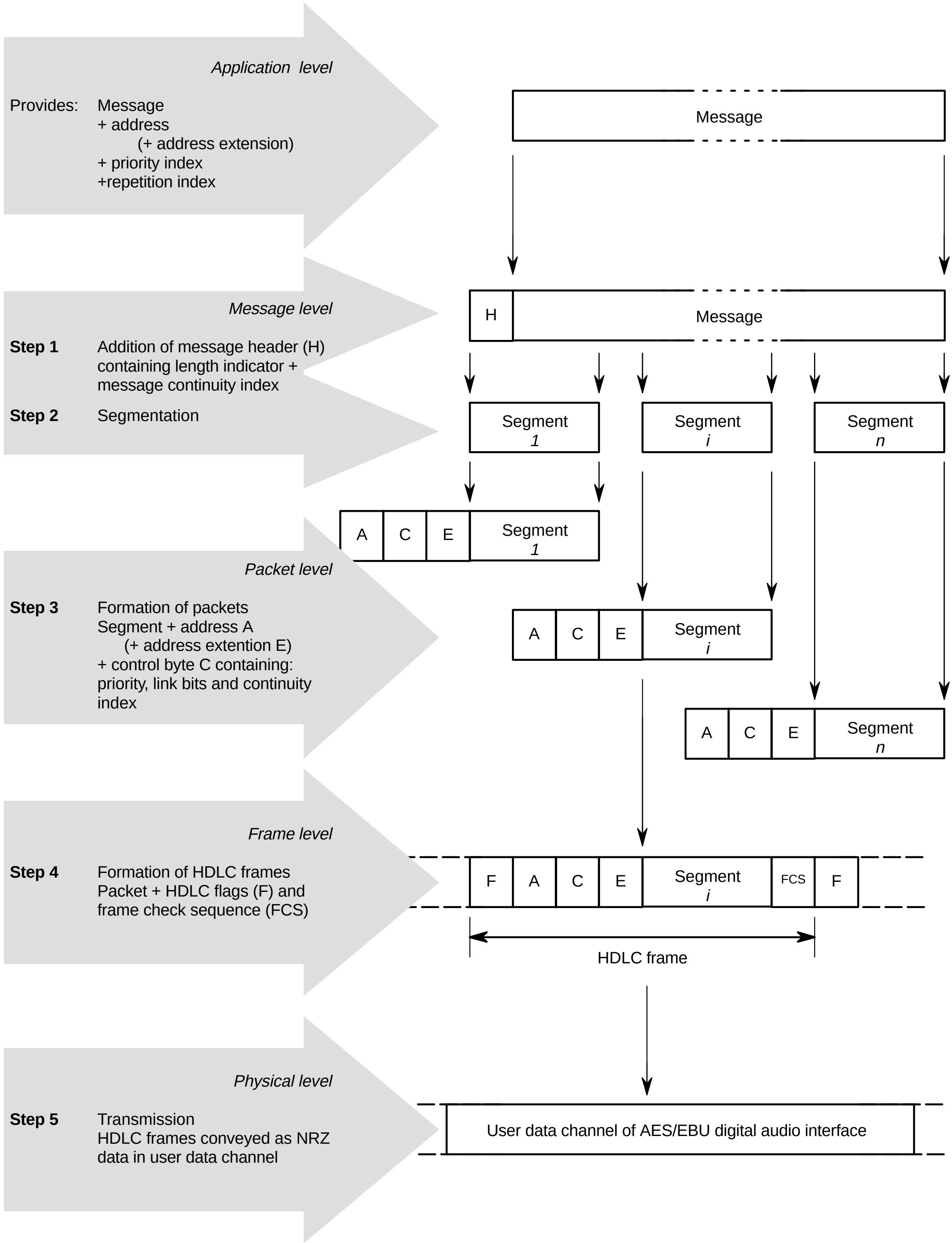

Fig. 1 – Main steps in the data formatting process.

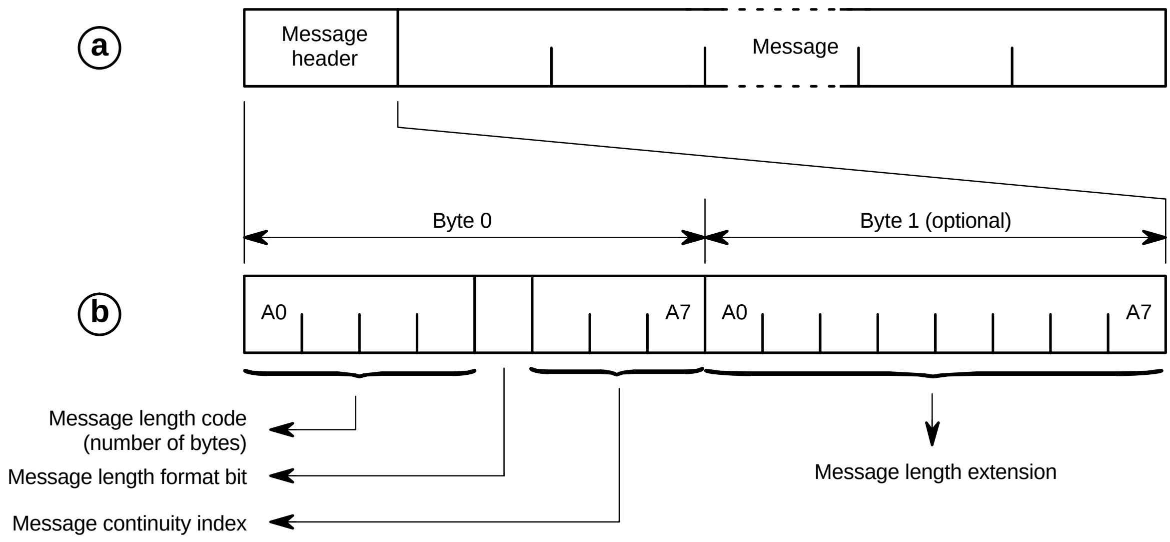

Fig. 2 – Message level
a) Message with header b) Message header

b) Message length coding

The messages can be any length and the different message lengths shall be coded in one of three ways according to the length. (*Note*: the message length does *not* include the message header byte or bytes.)

– If the length of the message is *less than or equal to 15 bytes*, the message header shall be one byte long since the message length can be given by the four–bit message length code (header byte 0, bits A3 to A0). The message length format bit (header byte 0, bit A4) is set to logic “0”.

– If the length of the message is *greater than 15 bytes but less than or equal to 4094 bytes*, the message length extension byte shall be used and the message length shall be coded by 12 bits (header byte 0, bits A3 to A0 plus the extension byte, header byte 1 A7 to A0), with the LSBs of the message length in the extension byte. The message length format bit (header byte 0, bit A4) shall be set to logic “1”.

– If the message length is *greater than 4094 bytes or unknown*, the message length shall be coded as 4095 (FFF_{hex}).

c) Segmentation

The message, complete with its header, shall be divided into segments of 16 bytes (or less if it is the last segment or only segment) (*Fig. 1* – step 2). The header shall always be part of the first segment.

4.2.2. Packet level

The packet level shall assemble the segments into packets (*Fig. 1* – step 3). From the message level it shall receive:

– the segments of the message;

– the parameters which were passed on from the application:

 – the address (and the address extension if there is one);

 – the priority index,;

 – the repetition index.

Each packet (*Fig. 3a*) shall be formed from one segment of the message with the addition of an address byte, a control byte and an address extension byte if present. The segment shall become the information field of the packet.

The segment, the address byte and the address extension byte, shall be inserted into the packet exactly as they are received from the message level.

a) *Control byte*

The control byte shall be generated at the packet level and shall be 8 bits long (A7,MSB to A0, LSB), divided into four fields (*Fig. 3b*).

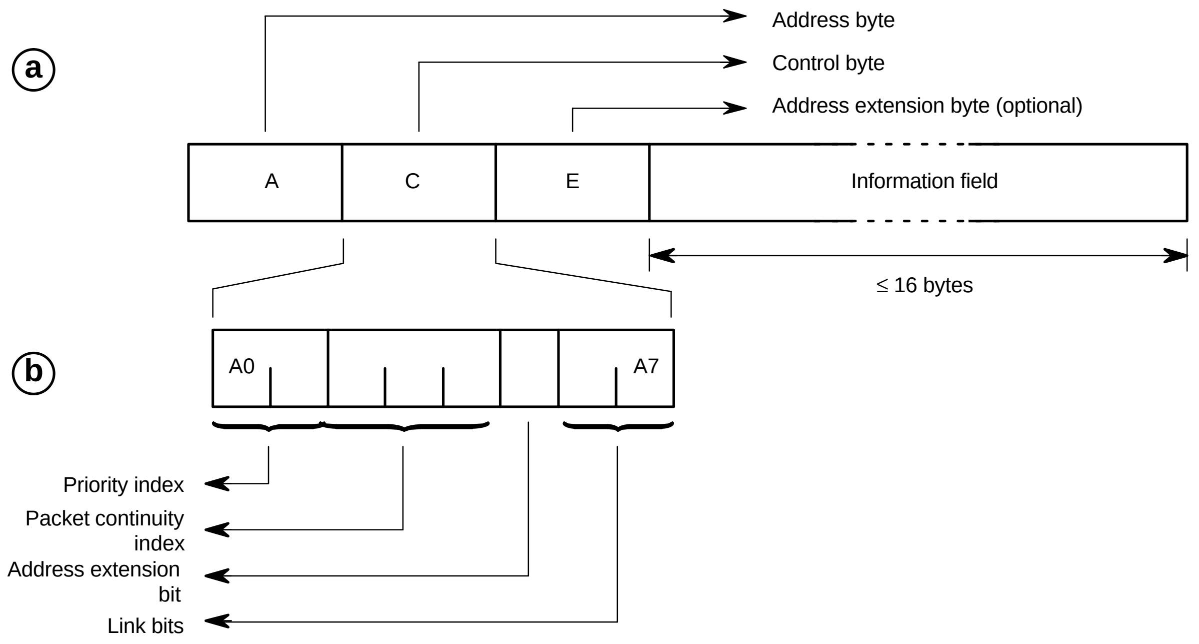

Fig. 3 – Packet level
a) Packet **b) Control byte**

Control byte (normal packets)		
Bits A0–A1		Priority index
		The priority indexshall be set by the application (see *Section 4.1.2.*).
Bit state	A0 A1	
	0 0	Priority index 0 (lowest)
	1 0	Priority index 1
	0 1	Priority index 2
	1 1	Priority index 3 (highest)
Bits A2–A4		Packet continuity index
		This index allows the receiver to detect whether a packet is missing from a message. It is sent with the same address as that defined by an application. When a packet is successfully repeated, under the control of the repetition index, the packet continuity index shall not change.
Bit A5		Address extension bit
	0	There is no address extension byte.
	1	There is an address extension byte.
Bits A6–A7		Link bits
Bit state	A6 A7	
	0 0	The packet is an intermediate packet of a message of three or more packets.
	1 0	The packet is the last packet of a message of two or more packets.
	0 1	The packet is the first or only packet of a message. (This packet always contains the message header.)
	1 1	Reserved for system packet identification (see *Section 5.2.1.*).

b) *Packet level repetition*

After the packets are formed, they shall be repeated as many times as defined by the repetition index and passed on to the frame level.

4.2.3. Frame level

The frame level shall receive the packets and construct the frames (*Fig. 1* – step 4) which are inserted into the user data channel. The frames shall use the High–level Data Link Control (HDLC) structure. The HDLC frame is defined in [2].

A frame (*Fig. 4*) comprises:

– beginning flag: 0111 1110 ($7E_{hex}$);

– packet;

– frame check sequence (a cyclic redundancy check);

– ending flag: 0111 1110 ($7E_{hex}$).

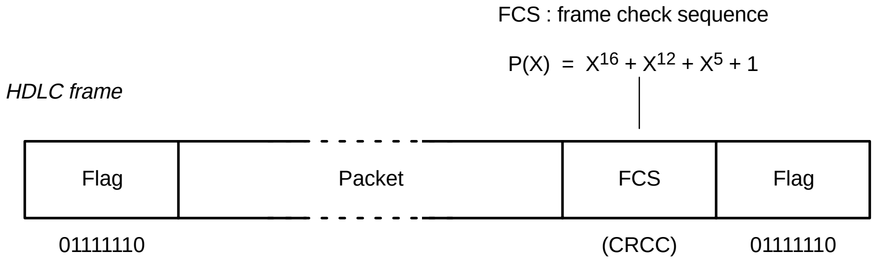

Fig. 4 – Frame level

When a series of packets is transmitted, the end flag of one frame may be the beginning flag of the next frame.

4.2.4. Physical level

The physical level (*Fig. 1* – step 5) shall be the serial transmission of the encoded frames at a bit–rate equal to the audio sampling frequency.

a) *Data coding*

The data in the packet part of each frame, including the FCS, shall be coded as described in [2] to insert one extra logic “0” after every sequence of five logic “1”s (*Fig. 5*). Sequences of six logic “1”s are therefore available to flag unambiguously the beginning and the end of a new frame, since this pattern cannot occur in data. The flag is the byte 0111 1110 (7E hex).

When there are at least seven consecutive logic “1”s the transport system shall be in the idle mode.

b) *Data transmission*

The frames shall be transmitted in the user data channel of the digital audio interface [1] as non–return–to–zero (NRZ) signals.

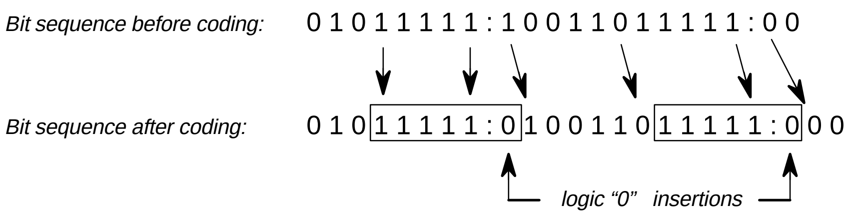

Fig. 5 – HDLC channel coding.

The use of this transport system shall be signaled in the channel status channel of the interface by setting channel status byte 1, bits 4–7 to code "0 0 1 0" (see Section 4 of [1]).

The least significant bit of each byte shall be transmitted first.

5. Channel management

The system described so far is able to transmit messages, but no special provision is made for inserting messages in downstream equipment or for resolving any conflicts of priority. The channel management system is provided to control these functions. It defines:

– a channel formatting and block structure;
– a channel description system;
– rules for inserting the data packets.

5.1. Channel formatting

The data flow in the user data channel shall be divided into repetitive structures called blocks. The block structure is shown diagrammatically in *Fig. 6*.

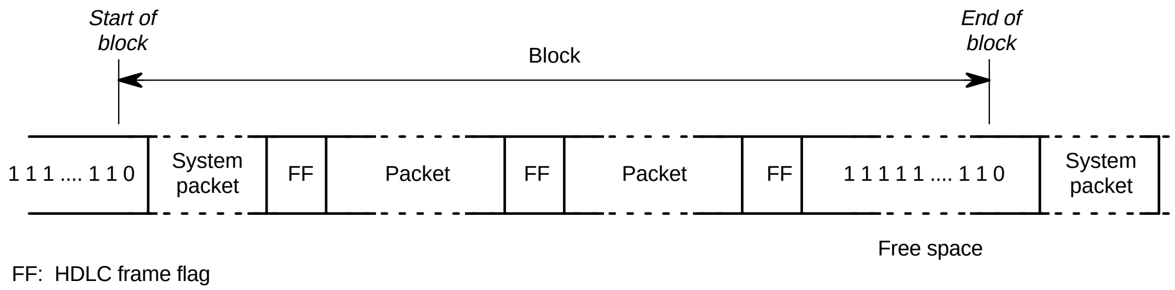

Fig. 6 – Block structure.

5.1.1. Block rate and length

The block structure can be synchronized to an external event and the user can set the repetition rate of the blocks to suit the application. Recommended repetition rates are listed in *Table 1*.

Any other repetition rate can, however, be defined by the user (see *Section 5.2.1.c*). These block rates are independent of the audio sampling frequency and will contain different numbers of bits at different audio sampling frequencies.

Table 1 – Recommended block repetition rates.

Blocks/s	Duration (ms)	Remarks
2	500	
5	200	
24	41.67	Film frame rate
25	40	PAL, SECAM or 1250/50 HDTV frame rate
29.97	33.37	NTSC frame rate
30	33.33	1125/60 HDTV frame rate
33.33	30	R–DAT frame rate
100	10	Shortest practical block

Normally, the number of bits in a block will be constant. However, in some cases it will be impossible to provide a constant number of bits in the blocks. For instance in the case of the NTSC frame rate and an audio sampling frequency of 48 kHz, there are 8008 audio samples every five video frames. Since the number 8008 is not a multiple of 5, it is necessary to have variable block lengths.

5.1.2. Block start / end

A block start shall identified by a logic “0” which follows at least seven logic “1”s. The block terminates with at least seven logic “1”s which are the logic “1”s before the first logic “0” of the following block. This logic “0” is defined as the first bit of the block and the HIGH to LOW transition which follows the sequence of logic “1”s is considered as the beginning of the block (*Fig. 6*).

5.2. Channel description

The channel shall be divided into blocks. The start of each block is indicated by the block start sequence described in *Section 5.1.2*.

Following the block start sequence an optional system packet may be added which describes the block length and controls the data insertion. The system packet can have an information field which can be used to carry system data.

5.2.1. System packet

The system packet shall contain an address byte, a control byte, an optional address extension, a descriptor byte and an information field, which can contain up to 15 bytes, (*Fig. 7a*).

The address FF_{hex} shall always used for the system packet and shall be restricted to the system packet to permit easy identification.

a) Control byte of system packets

The control byte of a system packet (*Fig. 7b*) shall have the structure defined below. This is a different structure to that of the control bytes previously described for normal packets (*Section 4.2.2.a*):

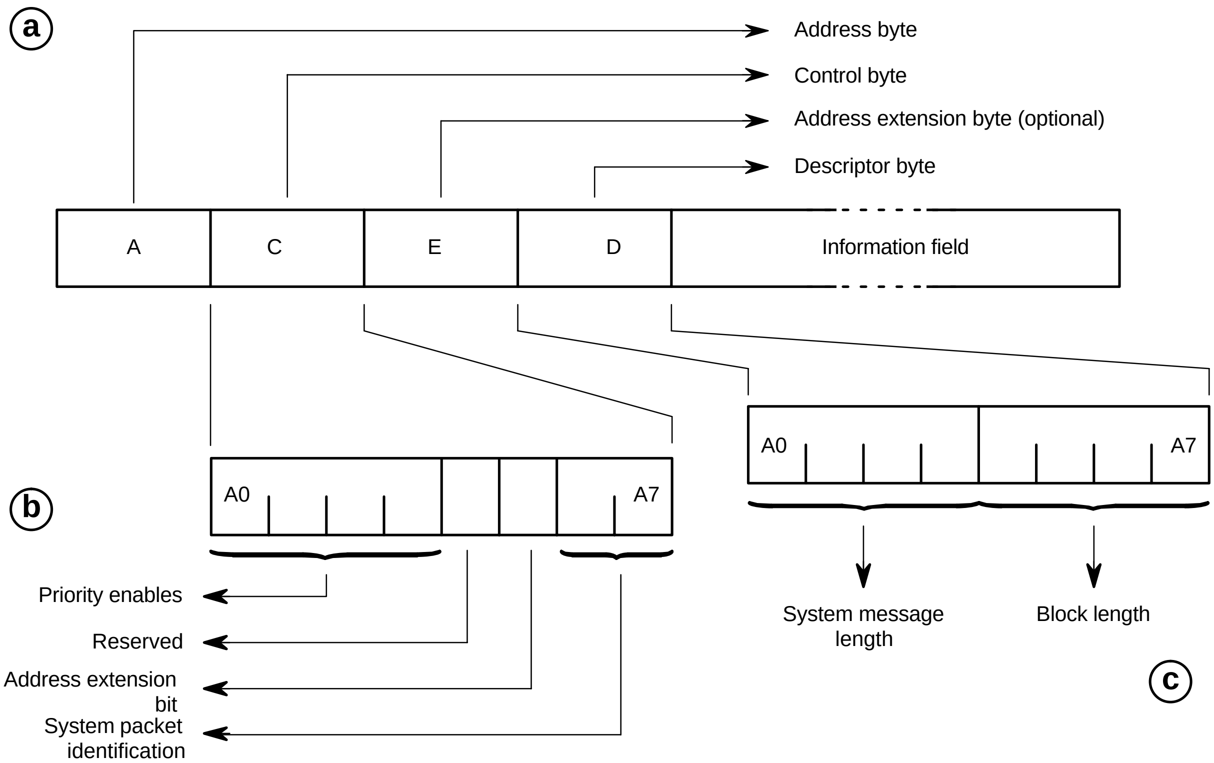

Fig. 7 – System packet
a) System packet **b) Control byte** **c) Descriptor byte**

Control byte (system packets)		
Bits A0–A3	Priority enable	
	When a priority enable bit is set to logic “1”, messages which have the corresponding priority can be inserted; when the bit is set to logic “0” the insertion is forbidden. There is no restriction on the combination of enable bits which can be set. For general rules of insertion see *Section 5.3.1*.	
Bit state	A0 A1 A2 A3	
	x	Priority enable for priority 0 (lowest)
	x	Priority enable for priority 1
	x	Priority enable for priority 2
	x	Priority enable for priority 3 (highest)
Bit 4	Reserved and shall not be used at present; set to logic “0”.	
Bit A5	Address extension bit	
	0	There is no address extension.
	1	There is an address extension (See *Section 5.2.1.b.*). Reserved, and shall not be used.
Bits A6–A7	System packet identification	
Bit state	A6 A7	
	1 1	This code is always used; it identifies the packet as a system packet.

b) Address extension byte

The address extension byte shall identify the contents of the information field in the system packet and shall only be transmitted when the address extension bit is set to logic “1”.

Address extension byte	
Bits A0–A7	Description of information field of system packet. Reserved for future use and set to logic “0”.

c) Descriptor byte

The 8–bit descriptor byte (*Fig. 7c*) shall have two fields which define the length of the block and the length of the system message.

Descriptor byte				
Bits A0–A3	System message length			
	Number of bytes in the information field of the system packet.			
Bits A4–A7	Coded length of block			
Bit states	A4 A5 A6 A7	*Blocks/s*	*Duration (ms)*	*Remarks*
	0 0 0 0	24	41.67	Film frame rate
	1 0 0 0	25	40	PAL, SECAM, 1250/50 frame rate
	0 1 0 0	30	33.33	1125/60 HDTV frame rate
	1 1 0 0	29.97	33.37	NTSC frame rate
	0 0 1 0	100	10	Shortest practical block
	1 0 1 0	5	200	
	0 1 1 0	2	500	
	1 1 1 0	33.33	30	R–DAT frame rate
	0 0 0 1	User–defined		
	All other states of bits A4–A7 are reserved and shall not be used.			

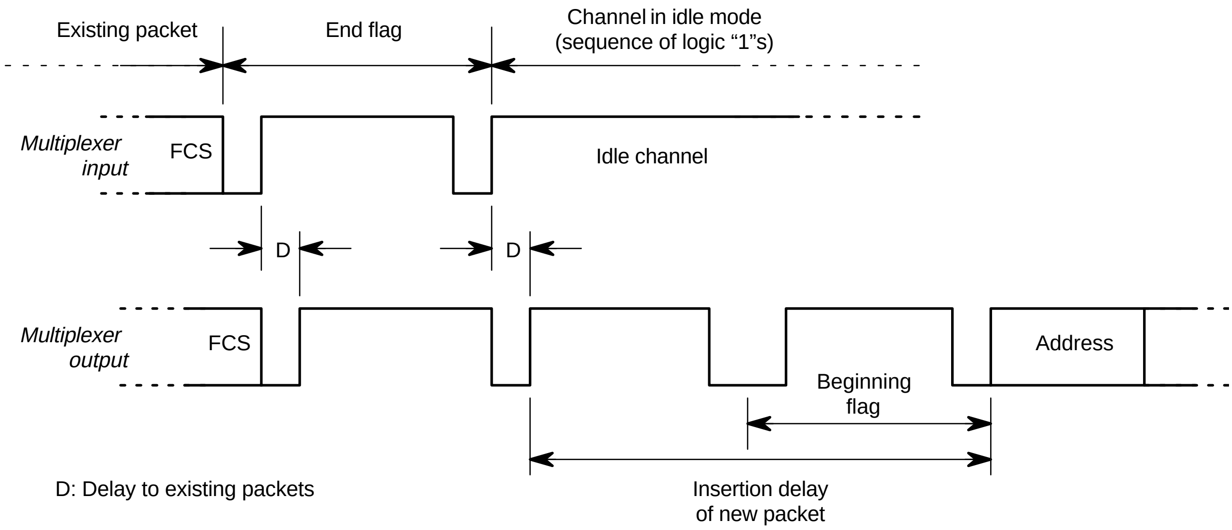

Fig. 8 – Insertion of a packet.

c) *Information field in the system packet*

The information field which follows the descriptor byte can contain up to 15 bytes of data (*Fig. 7*). It can be used to carry time codes, to indicate the block length when it is user defined, or to carry other information. The content of the information field, if present, shall be identified by the address extension byte of the system packet (see *Section 5.2.1.b*).

Note: The use of the information field awaits definition of the address extension byte.

5.3. Data insertion

Packets shall be inserted into the blocks of the user channel by packet multiplexers. When packets are due to be inserted, a block may already contain a number of packets. Existing packets will be followed by successive ONES which indicate an idle channel up to the end of the block. The ending flag of a packet and the beginning flag of the following packet can be the same, or can be separated by up to three bytes.

5.3.1. Rules of insertion

The multiplexing of new packets involves two steps. The first step is the detection of a block start (see *Section 5.1.2.*). The second step is the detection of seven consecutive logic "1"s which indicate that the channel is in the idle mode. This allows the multiplexer to insert a new packet if there is enough free space before the end of the block. If there is space for the new packet, the seventh existing logic "1" shall be set to logic "0", forming a new HDLC flag. The new packet (possibly including a further HDLC "beginning" flag) shall then follow (*Fig. 8*).

Table 2 – Examples of numbers of justification bits for various block lengths.

Sampling frequency kHz	Free space %	Total bits per block			Justification bits per block		
		10 ms	40 ms	200 ms	10 ms	40 ms	200 ms
42	0	420	1680	8400	0	0	0
44.1	5	441	1764	8820	21	84	420
48	**12.5**	**480**	**1920**	**9600**	**60**	**240**	**1200**
54	25	540	2160	10800	120	480	2400

The amount of available free space depends upon the block length and the length of previously–inserted messages. However free space (justification bits) shall be reserved at the end of each block to avoid losses of packets during sampling frequency conversion or variable speed operation. The amount of free space which shall be reserved should allow the audio sampling frequency to be reduced to 12.5 % below 48 kHz. Examples are given below for various audio sample rates and various block lengths.

The use of justification shall be mandatory over this audio sampling frequency range. The interface and the transport system can be used at other audio sampling frequencies, for instance 32 kHz, but will need appropriate data management.

5.3.2. Priority management

Priority management shall be carried out at the multiplexing level and, in the first instance, the insertion of messages shall be allowed or forbidden according to the state of the priority enable bits of the system packet (see *Section 5.2.1.a*). Only packets having a priority whose priority enable bit is set at logic "1" shall be inserted.

The priority management rules further ensure that the multiplexers are able to average the loading of the blocks to prevent upstream multiplexers monopolizing the channel capacity.

When the application is designed, the user shall assign a priority to each message. This priority level may be chosen either:

– to ensure a minimum delay in the insertion of short urgent messages carrying real–time data;

or

– to provide a sufficient bit–rate, compatible with the application, if the messages are long.

a) Priority insertion rules

The priority management rules shall be in accordance with *Table 3*. The table defines the maximum number of packets which can be inserted per block as a function of the block length and the priority.

When the number of packets which can be inserted into a block for a given priority is greater than or equal to one, packets can be multiplexed according to the rules defined in *Table 3*.

Table 3 – Maximum number of packets per message inserted per block, as a function of block length and priority.

Priority	Block length			
	10 ms	1 frame	200 ms	500 ms
3 (highest)	1	4	20	50
2	1/4	1	5	12
1	1/20	1/5	1	2
0 (lowest)	1/40	1/10	1/2	1

However, for long messages of lower priorities, where the rules will only allow one packet to be inserted into one of several blocks, extra rules are needed to prevent overloading of the earlier blocks. In these circumstances the following rules shall apply in order to spread the loading over all the available blocks.

If the insertion rules allow one packet to be introduced into every n blocks (n = 2, 4, 5, 10, 20, 40, defined in *Table 3*), then:

– the packet shall only be inserted into the first $n/2$ blocks if there is at least one of the blocks with more than half of its length still available for downstream multiplexing;

– If the packet is not inserted in the first $n/2$ blocks, it shall be inserted into the earliest–available block.

b) Real–time applications

For urgent or critical messages, which are usually one packet long, the priority index can be used to control the insertion delay of the message packets. The block length represents the maximum delay needed to multiplex a packet into the network (a single packet message can be inserted into a single block). The block length will be chosen to give a delay which is appropriate to the application.

Table 4 shows how the maximum delay varies with the block lengths and priority.

Table 4 – Maximum delay of packet insertion, as a function of block length and priority.

Priority	Block length			
	10 ms	1 frame	200 ms	500 ms
3 (highest)	10 ms	1 frame	200 ms	500 ms
2	1 frame	1 frame	200 ms	500 ms
1	200 ms	200 ms	200 ms	500 ms
0 (lowest)	200 ms	500 ms	500 ms	500 ms

c) Bit–rate adaptation

The priority management rules and the choice of priority can be used to determine the number of packets that can be inserted into one block and hence the effective bit–rate (see *Table 5*). For example, for a 10 ms block length and a message with priority 3 (the highest) one packet will be inserted into every block during the transmission. However with the same block length but a message with priority 2, one packet will be inserted into only one block out of every four blocks. The mean value is one packet per 40 ms.

Table 5 – Priority management rules.

Priority	Approximate packets/s	Approximate bits/s
3 (highest)	100	12800
2	25	3200
1	5	640
0 (lowest)	2.5	320

Bibliography

[1] *Specification of the digital audio interface (The AES/EBU interface)*
EBU document Tech. 3250, 2nd edition, 1992.

[2] *Information processing systems – Data communication – High–level data link frame structure.*
ISO Publication 3309–2

Note: other versions of the specification of the digital audio interface exist:

Digital Audio Interface
IEC Publication 958, 1st edition 1989

AES Recommended practice for digital audio engineering – Serial transmission format for two–channel linearly represented digital audio data
AES3–1992.

A digital audio interface for broadcasting studios.
CCIR Recommendation 647.

Annex
Data management guidelines

This Annexe *is not part of the user data channel specification defined in this document. It is gives additional information concerning management of the data capacity available in the user data channel.*

Provided that sensible data management is performed to remove redundant data, the system capacity is such that it is unlikely that a channel will be fully–loaded for a sufficiently long period as to prevent the insertion of new packets according to the rules described in *Section 5* of this document. If this situation does arise, two options are available to enable new packets to be inserted:

– Existing packets which are no longer required can be deleted and the remaining packets re–ordered to create sufficient space for new packets to be inserted. The number of new packets that may be inserted will depend upon the amount of free capacity created and the length of the packets to be inserted. The deletion and re–ordering process requires that packets be removed from one block and then re–inserted in the following block, thereby resulting in a one–block delay. All subsequent packets in the channel will experience the same delay until an empty block is found.

– Existing packets can be over–written, but in this case there is the possibility of losing data which may be wanted. Depending on the length of the packet to be inserted, and the lengths of the existing packets in the channel, it is possible that more than one packet will be over–written. The new packet to be inserted must be preceded by at least one HDLC start flag.

Editeur responsable : G.T. Waters, Ancienne Route 17A, CH–1218 Grand–Saconnex/Genève, Suisse.