

**ISMA  
INTERNET STREAMING MEDIA ALLIANCE**

Contribution Registration Number **TN01082**  
**February 2005**

**TITLE: Support for Streaming Text and Line 21 Data Services in MPEG-4  
Streaming Applications**

**Editor: Sassan Pejhan (VBrick Systems, sassanp@vbrick.com)**

**AREA / Task Force: Closed Captioning**

**Status: Internal Draft Specification v.0.6**

# Table of Contents

<a href="#">1</a>	<a href="#">Introduction</a>	2
<a href="#">2</a>	<a href="#">Marketing Requirements</a>	2
<a href="#">3</a>	<a href="#">Technical Details and Specifications</a>	2
<a href="#">3.1</a>	<a href="#">Changes to SDP file</a>	4
<a href="#">3.2</a>	<a href="#">Carriage of Line 21 Data over RTP</a>	5
<a href="#">3.2.1</a>	<a href="#">RTP packet structure for Line 21 data payload</a>	5
<a href="#">3.3</a>	<a href="#">Carriage of 3GPP Timed Text over RTP</a>	7
<a href="#">3.4</a>	<a href="#">Storage and Signaling of Line 21 data in MP4 file</a>	7
<a href="#">3.4.1</a>	<a href="#">Media Handler</a>	8
<a href="#">3.4.2</a>	<a href="#">Media Information Header</a>	8
<a href="#">3.4.3</a>	<a href="#">Data Reference ‘dinf’ and ‘dref’ boxes</a>	8
<a href="#">3.4.4</a>	<a href="#">Sample Size ‘stsz’ box</a>	8
<a href="#">3.4.5</a>	<a href="#">Sync (Random access) sample ‘stss’ box</a>	8
<a href="#">3.4.6</a>	<a href="#">Sample Description Format</a>	8
<a href="#">4</a>	<a href="#">Other Issues</a>	9
<a href="#">5</a>	<a href="#">References</a>	9
<a href="#">Appendix A: Other Approaches</a>		10
<a href="#">MPEG-4 BIFS</a>		10
<a href="#">User data in Video Stream</a>		10

# 1 Introduction

This document defines a method for supporting Line 21 data services as specified in [1] and other streaming text specifications (such as 3GPP Timed Text[8]) in real-time, ISMA-1.0 streaming applications. Line 21 data services include Closed Captioning (CC), but also Text Services and Extended Data Services (XDS). A key requirement for the digital set-top box and digital television market is support for Closed Captioning (CC). While the emphasis of this document is on EIA/CEA 608-B specified data, the scheme allows the support of other, non-MPEG-4 standards-based streams (e.g. EIA 708).

## 2 Marketing Requirements

There are several reasons why support for Closed Captions within ISMA 1.0 compliant streams is required. First, there is an obvious human need to deliver visual information for people with hearing disabilities. Second, there is a US Government mandate which states that “*All training and informational video and multimedia productions which support the agency's mission, regardless of format, that contain speech or other audio information necessary for the comprehension of the content, shall be open or closed captioned*” (US 29 USC 794d). Finally, popular proprietary streaming formats currently support the end-to-end delivery of text messages, which are used to deliver Closed Captions, and not supporting Closed Captions would put ISMA compliant systems at a serious competitive disadvantage.

## 3 Technical Details and Specifications

The primary technical objective is to carry the text stream as an *opaque* bitstream, synchronized, and along, with other MPEG-4 streams (primarily audio and video). Thus the stream need not be converted or interpreted. This document defines the mechanisms to:

- Signal the presence of this additional stream as part of an MPEG-4 streaming application, and its association with MPEG-4 audio and video streams;
- Transport this additional stream over RTP in a manner that allows synchronization with the MPEG-4 audio and video streams. In this document, the transport over RTP is defined specifically for EIA/CEA 608-B data. Transport of 3GPP Timed Text over RTP is being defined by the IETF [9]. Similar specifications will have to be defined for other (e.g. EIA 708-B) data.
- Store the data from this stream as a separate track in MP4 files.

New subtypes of the ‘text’ MIME type are also defined as identifiers for the various 3<sup>rd</sup> party standards bitstreams. This will facilitate interoperability between set-top boxes

from different manufacturers – many of whom already provide support for EIA/CEA-608-B specified data.

Receivers that do not require the opaque data stream can simply ignore it or specifically signal the sender not to send it (unicast applications).

A benefit of the proposed scheme is that a set-top box can simply take the Line 21 data off the network, convert back to analog (video D/A chips usually have this capability) and send it directly to a TV. The decoder chip in the TV does the work of converting the data to text on the screen. If a software player on a PC would like to show the Closed Captioning or other Line 21 data (such as XDS), it will need to implement a Line 21 data decoder in software using the rules described in EIA/CEA-608-B (freely available Line21 decoders are currently available from software vendors).

Use of a third stream has a number of advantages:

- Clients can select whether or not they receive the third stream.
- Clients can select to receive only the third stream (without video)
- The additional data does not have to be inserted into the compressed video bitstream (for hardware based encoders, this implies that the encoded video bitstream need not be parsed)
- The scheme is extensible and can accommodate other standards-based bitstreams.

The disadvantages of having a third stream are:

- Increased bandwidth usage due to the RTP/UDP/IP overhead associated with a third stream
- Extra CPU cycles needed to support a third stream (additional stack work, additional sender reports, etc.)
- Changes need to be made to the SDP file
- An additional track (or two for hinted files) will be added to the MP4 file
- A method for packing multiple data samples into a single RTP packet is required to reduce stack overhead (text 'frames' or access units are typically only a few bytes in size).

Two mechanisms need to be defined to support a third stream:

1. Signal and identify the stream. This is primarily accomplished by adding a new media section to the SDP file.
2. Transport and Synchronization of the stream. This is accomplished by opening a third pair of RTP/RTCP connections between server and client.

The details of these mechanisms, as well as storage in MP4 files, are discussed in the subsections below.



a=control:trackID=<id>

## 3.2 Carriage of Line 21 Data over RTP

This section describes mechanisms for carrying EIA/CEA-608-B data over RTP. Similar mechanisms can be developed for other 3<sup>rd</sup> party standards-based data.

Either two or four bytes of Line 21 data can be generated for each video frame. RTP packet headers, on the other hand, are at least 12 bytes in length. If only one Line 21 'sample' is included in one RTP packet, the additional bit-rate, including the UDP and IP headers, would be at least  $(4+12+8+20)*30*8 = 10,560$  bps. That may be negligible when transmitting over broadband connections, and a price worth paying given the simplicity and low latency of the scheme.

On the other hand, at low bit-rates it would be wasteful to use 42/44 bytes to convey only two/four bytes of information. In this situation, it would be much more efficient to pack several Line 21 'samples' or Access Units (AU) into one RTP packet. Reducing the overall number of RTP packets will also help reduce the CPU load imposed by the network stack in an embedded environment.

Given the small size of Line 21 data, it is theoretically possible to pack 375 samples into one RTP packet for a network with a 1500 byte MTU. Line 21 data is time sensitive, however. Packing too many samples into a single packet will lead to high latency (375 samples at 30 fps would incur more than 12 seconds of latency!). The number of samples packed into an RTP packet will in practice be limited by the maximum latency allowed.

The solution proposed here allows the number of Line 21 data samples packed into an RTP packet to be variable.

### 3.2.1 RTP packet structure for Line 21 data payload

The ISMA specifications provide a detailed mechanism for concatenating multiple access units in a single RTP packet (see section 3.2.1 of Appendix G in [2]). This mechanism is overkill for Line 21 data, however, and exacerbates the excessive overhead problem. A simpler mechanism is therefore provided below.

The proposed mechanism assumes the use of the standard, 12-byte RTP header. The structure of the RTP packet is shown in Table 1 below:

Field	Length (bits)	Semantics
RTP packet header	96	Standard RTP header
Flags {		
Version	2	Version of this specification

Reserved	6	Reserved for future use
}		
Access Units	AccessUnitsPerPacket x 40	

**Table 1** Structure of RTP packet

The RTP marker bit ‘M’ is always set to 1. Access Unit fragmentation is a moot point, given the tiny size of Line 21 Access Units.

The 12-byte RTP header is followed by a one-byte **Flags** field, which in turn has two subfields.

**Version** – This field identifies the version of the specification of the RTP packet structure for carrying Line 21 data, enabling forward compatibility with future versions. It should be set to zero (00) for this document.

**Reserved** – These six bits are reserved for future use. In the current scheme, they will be used as padding to achieve byte alignment. It is recommended that they all be set to 0.

The **flags\_byte** parameter in the SDP file (see section 3.1 above) shall be identical to the first byte in each Line 21 RTP packet.

The actual Line 21 data access units will then follow immediately after the **Flags** field. Each Access Unit will be exactly 5 bytes long. These will be concatenated in presentation order and will follow the first field. The number of access units in each packet can be variable. The receiver can determine this number from the size of the RTP payload:

$$\text{Number of CC AUs} = (\text{Payload size} - \text{Flags byte})/5.$$

The structure of the 5-byte AU field will be as shown in Table 3. The **cc\_valid** field is identical to the one proposed by the ATSC in [3]. It will indicate whether the Line 21 data bytes are valid or not. Some hardware-based encoders only implement Field 1 Line 21 data services. They therefore do not generate valid Field 2 Line 21 data. Similarly, some encoders may only use Field 2 Line 21 data (e.g. for XDS) and not generate any CC or Teletext services for Field 1. In these situations, the appropriate flag should be set to ‘0’ (invalid). Otherwise, they shall be set to ‘1’.

Field	Length (bits)	Semantics
cc_valid_1	1	Flag indicating if first two data bytes are valid
cc_valid_2	1	Flag indicating if second two data bytes are valid
Padding	6	Required to make Line 21 AU byte aligned
Field1_data1	8	First byte of Field 1 Line 21 data
Field1_data2	8	Second byte of Field 1 Line 21 data

Field2_data1	8	First byte of Field 2 Line 21 data
Field2_data2	8	Second byte of Field 2 Line 21 data

**Table 2 Access Unit Syntax and Semantics**

All four Line 21 bytes are always present. When a flag is set to '0', it is recommended, though not required, that the corresponding Field<x>\_data<x> bytes be set to 0x00.

The RTP Timestamp must carry the sampling instance of the first Line 21 Access Unit. The timestamp of subsequent Line 21 access units can be calculated from this timestamp, the RTP timescale and the frame rate (that may explicitly be stated in the SDP information or derived from the MIME sub-type). Note that since Line 21 data is present for all frames (regardless of whether the video encoder transmits the corresponding video frame or not) this scheme is not affected by the potential VFR nature of the digitized, MPEG-4 encoded video stream.

Should the decoder/receiver detect that a packet has been lost by the network, it may have to generate NULL Line 21 AUs (0x80) in place of the missing ones (depending on the nature of the receiving/display device). Since the number of AUs per packet is variable, the client would calculate the number of AUs lost based on a) the timestamps of the packets immediately preceding and immediately succeeding the lost packet; b) the number of AUs in the packet immediately preceding the lost one; and c) the 'frame-rate' of the Line21 stream (30000/1001).

The specified scheme for carriage of Line21 over RTP has the following characteristics:

- There is Line 21 data for all frames, and at a fixed frame rate.
- The size of RTP packets is  $12+1+n*5$  where  $n$  is the number of AUs in the packet.
- The timestamp of the first AU in a packet is signaled explicitly; timestamps of the rest of the AUs are derived and not signaled explicitly.

### **3.3 Carriage of 3GPP Timed Text over RTP**

The IETF is currently defining the specifications for carrying 3GPP Timed Text over RTP. The latest version of these specifications is provided in [9]. In keeping with ISMA practice, we will use specifications from other standards bodies where possible.

### **3.4 Storage and Signaling of Line 21 data in MP4 file**

What has been proposed above is sufficient to allow transmission of Line 21 data from a live source over the network and to a receiver for playback. Further mechanisms are required to enable storage and retrieval of text streams such as Line 21 data to/from an MP4 file.

### 3.4.1 Media Handler

While the `DecoderConfigDescriptor` allows for the signaling of user private streams, the MP4 file format [7] – which is derived from the ISO base media file format [6] – does not explicitly support the storage of user private streams. Specifically, the `handler_type` field in the track headers in an MP4 file is limited to a list of 10 types [6,7], none of which is appropriate for user private or other non-MPEG data.

The 3GPP Timed Text specifications [8] propose the additional handler type ‘text’. This will be the handler type used for the various ‘text’ streams defined in this document.

### 3.4.2 Media Information Header

A Null Media Header box (‘nmhd’) as defined in [6] shall be used in the Media Information Header Box of the third stream.

```
aligned(8) class Mpeg4MediaHeaderBox
    extends FullBox(‘nmhd’, version = 0, flags) {
}
```

flags is a 24 bit integer - currently all bits are set to 0.

### 3.4.3 Data Reference ‘dinf’ and ‘dref’ boxes

These shall be identical to those of audio and video (files will be self-contained per ISMA restrictions).

### 3.4.4 Sample Size ‘stsz’ box

This table lists the size of each individual access unit in a stream. When all the access units are of the same size - as will be the case with Line 21 data - a compact form can be used where a single default sample size is stored.

### 3.4.5 Sync (Random access) sample ‘stss’ box

This box lists all samples that are random access points. It is optional. When absent, it implies that all samples are sync points. Since Line 21 samples can be ‘decoded’ independently, we will consider all of them as random access points, and hence omit this table.

### 3.4.6 Sample Description Format

The sample description format for 3GPP text is specified in [8], and uses the ‘tx3g’ tag to identify the sample format. For Line 21, a similar, though simpler format is proposed:

The sample description format for Line 21 is as follows:

```
class TextSampleEntry() extends SampleEntry ('ln21') {
    unsigned int(8)      flagsbyte;
}
```

flagsbyte is the same byte that appears in the SDP file and is present at the beginning of each Line21 RTP packet.

## 4 Other Issues

Integration with ISMAcrypt has not been addressed here. Issues that remain open are:

- Extension of scheme to support teletext. In some markets, there is a demand to support teletext, which is similar to CC but with a higher data rate. Teletext streams can be signaled in the SDP file as illustrated above. A detailed mechanism for transmission of teletext data over RTP, similar to the one given above for EIA/CEA 608-B data, needs to be developed.
- Similarly, a mechanism for transmission of Extension EIA 708 CC for DTV over RTP needs to be developed. Signaling 708 streams in SDP has already been discussed.

## 5 References

- [1] *Line 21 Data Services*, EIA/CEA 608-B.
- [2] Internet Streaming Media Alliance Implementation Specification, Version 1.0, 28 August 2001.
- [3] *ATSC Standard A/53B with Amendment 1*, “ATSC Standard: Digital Television Standard, Revision B, with Amendment 1” (available at <http://atsc.org/standards.html>).
- [4] *Coding of Video and Associated Audio, ISO/IEC 14996-1*, N5065, August 2002.
- [5] RFC 3550, *RTP: A Transport Protocol for Real-Time Applications*, July 2003.
- [6] *Coding of audio-visual objects – Part 12: ISO base media file format, ISO/IEC 14996-12:2003(E)*, N5295, March 2003.
- [7] *Coding of Audio, Picture, Multimedia and Hypermedia Information – Part 14: MP4 File Format, ISO/IEC 14996-14:2003(E)*, N5298, March 2003.

[8] *3GPP TS 26.245 V1.0.0*, "Transparent end-to-end packet switched streaming service (PSS); Timed text format (Release 6)", May 2004.

[9] *draft-ietf-avt-rtp-3gpp-timed-text-11.txt*, "RTP Payload format for 3GPP Timed Text", January 2005.

## **Appendix A: Other Approaches**

Two other approaches for supporting Closed Captioning were also considered but were discarded in favor of the approach described above. These are discussed below for the sake of completeness.

### ***MPEG-4 BIFS***

MPEG-4 provides support for CC through the BIFS scene description stream. Both the textual content of CC and the font and layout information are included in BIFS. The BIFS stream can be synchronized with the video stream on a frame-by-frame basis, using the same clock reference. MPEG is currently defining extensions to the text format to allow for a more predictable rendering both in terms of fonts support and layout management. BIFS can also support most (if not all) of the services provided by the other types of Line 21 data (such as text services and XDS).

The most obvious drawback here is that we can no longer treat the Line 21 data as opaque, since it has to be converted into BIFS syntax. This conversion process will consume CPU cycles, and will negatively impact real-time performance. We also have to send BIFS update commands. Since BIFS is about much more than text and layout information, receivers will also have to implement full support for BIFS streams. This goes against ISMA's philosophy of taking a minimalist approach to MPEG-4 systems. It is also not clear if *all* non-CC data in Line 21 (such as XDS) can be supported via BIFS.

### ***User data in Video Stream***

Another alternative is to do exactly what is done in the MPEG-2 systems: insert the Line 21 data in the `user_data` field of the MPEG-4 Video Elementary Stream. The precise method and syntax for including EIA/CEA 608-B data in MPEG-2 is specified by the ATSC [3].

A big caveat with this scheme is that while MPEG-2 allows the insertion of a `user_data` field for every frame, MPEG-4 allows it only at the much coarser levels of a Group of Video Object Planes (GOV), Video Layer, Video Object and Visual Sequence. This means that CC or other, potentially time sensitive data, can only be sent in batches and at time intervals that could be several seconds long.

ATSC specifications for inclusion of Line 21 data impose a 96-bit overhead – in addition to the 32-bit `user_data_start_code` and the Line 21 data itself – per frame. While this overhead and associated Line 21 data was negligible at the operating bandwidth of most MPEG-2 applications, they could become significant at the much lower bandwidths associated with MPEG-4 streaming. It is, of course, possible to use only a small subset of the ATSC structure to reduce the overhead.

Another drawback of this scheme is that bandwidth limited receivers which do not interpret Line 21 data will experience poorer quality video than they would with the proposed scheme, where they could simply ask the sender to not send the 3<sup>rd</sup> opaque data stream.

Moreover, most vendors of DSP/chip-based MPEG-4 video encoders currently do not insert Line 21 data into the `user_data` field. This would mean that manufacturers of encoder appliances would have to parse the (variable-sized) VOP headers and insert the Line 21 data in the `user_data` field on the fly. This has a negative impact on real-time operation.

Also, insertion of Line 21 data will become tricky under Variable Frame Rate scenarios, where frames can be dropped to achieve a certain video bit-rate target. Line 21 data must either be dropped, or multiple Line 21 samples – along with their individual timestamps – need to be inserted in the next transmitted VOP.

The advantages and disadvantages of the `user_data` field approach can thus be summarized:

Advantages:

- No RTP/UDP/IP overhead
- No extra stack work
- No need to change SDP (although we could make a change to signal Line 21 type in SDP)
- No hinting for Line 21 stream
- No extra file track

Disadvantages:

- Since `user_data` is not available on a per VOP basis, it becomes very risky
- VFR scenarios are tricky because Line 21 data is either dropped, or multiple data from different time instants have to be packed and signaled in the same field
- Line 21 data overhead will reduce video bandwidth (1.2 kbps with modified ATSC structure)
- Require parsing at both encoder and decoder to insert/extract Line 21 data
- Line 21 type cannot be signaled at Session level (though we can add a line to the video information in the SDP file to make this so)