Digital Audio-Visual Council

DAVIC 1.5 Specification

Jitter concealment tools

Technical Tool Specification

(Provisional Document Structure)

Revision 1.0

This revision 1.0 of the DAVIC 1.5 Specification Jitter concealment tools document supersedes all previous versions.

Date: 1999-01-22

Published by Digital Audio-Visual Council

Geneva, Switzerland

CONTENTS

Foreword

This document is part of DAVIC 1.5 Specifications.

The new series of specifications comprises several types of documents : specification of tools, system design an architecture specifications, service requirement specifications, service implementation guidelines (contours) and association of tools for this implementation.

This document contains a tool specification related with synchronisation problems in the context of MPEG2 used on IP based networks.

1 Scope

The schemes described in this document are intended to address the problem of clock synchronization, and decoder's buffer overflow or underflow in typical MPEG decoding operations. Implicit in the schemes are the assumptions that the network bandwidth requirements are appropriately satisfied (e.g. through RSVP), and that the decoder's clock frequency of 27 MHz is sufficiently accurate for chroma sub-carrier recovery. The schemes are identified as valid DAVIC tools, but none of the schemes has been mandated so far. Decisions to mandate any of the schemes may be made in the subsequent revisions.

2 References

2.1 InformativeReferences

[1] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[2] H. Schulzrinne, "RTP Profile for Audio and Video Conferences with Minimal Control", RFC 1890, January 1996.

[3] D. Hoffman, G. Fernando, V. Goyal and M. R. Civanlar, "RTP Payload Format for MPEG1 / MPEG2 Video", draft-ietf-avt-mpeg-new-01, Internet draft, June 1997.

[4] ISO/IEC International Standard 13818-1, "Generic Coding of Moving Pictures and Associated Audio Information: Systems", July 1995.

[5] Jacobson, V., "Congestion Avoidance and Control", Proc. SIGCOMM ’88 Conf., ACM, pp. 314-329, 1988.

3 Definitions

Jitter The statistical variance of the data packet interarrival time.

4 Acronyms and abbreviations

AV	Audio/Visual
CCRC	Contributing Source
DTS	Decoding Time Stamps
FIFO	First In First Out
IP	Internet Protocol
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6 (IP New Generation)
ISO	International Standardization Organization
ITU	International Telecommunications Union
LAN	Local Area Network
MPEG	Moving Pictures Expert Group
PC	Personal Computer
PCR	Program Clock Reference
PLL	Phase-Lock Loop
PTS	Presentation Time Stamps
QoS	Quality of Service
RQRP	Request/Reply Protocol
RSVP	Resource reSerVation Protocol
RTCP	Real Time Control Protocol
RTP	Real-time Transport Protocol
RTSP	Real-Time Stream control Protocol
RTT	Roundtrip Time
SCR	System Clock Reference
SDP	Session Description Protocol
SSRC	Synchronization Source
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
TS	Transport Stream
TV	Television

5 Conventions

The style of this Specification follows the Guide for ITU-T and ISO/IEC JTC1 co-operation. Appendix H: Rules for presentation of ITU-T | ISO/IEC common text (March, 1993).

6 Introduction

Real-time audiovisual applications using MPEG often do not work well across the Internet because of insufficient bandwidth, latency, variable queuing delays and congestion losses. The Internet, as originally conceived, offers only a very simple quality of service (QOS), point-to-point best-effort data delivery. This document contains an overview of the problems that could arise as a result of variable delays (jitter), and has assembled a number of schemes and mechanisms which could be applied under different conditions to overcome the network jitter problem.

6.1 An Overview of the Problem[1]

MPEG Systems embodies a timing model in which all digitized pictures and audio samples that enter the encoder are presented exactly once each, after a constant end-to-end delay, at the output of the decoder as they are the encoder. There is a single, common system clock in the encoder, and this clock is used to create timestamps that indicate the correct presentation and decoding timing of audio and video, as well as to create timestamps that indicate the instantaneous values of the system clock itself at sampled intervals. The timestamps that indicate the presentation time of audio and video are called *Presentation Time Stamps* (PTS). Those that indicate the decoding time are called *Decoding Timestamps* (DTS), and those that indicate the value of the system clock are called the *System Clock Reference* (SCR) in Program Streams and the *Program Clock Reference* (PCR) in Transport Streams. It is the presence of this common system clock in the encoder, the timestamps that are created from it, and the recreation of the clock in the decoder and the correct use of the timestamps that provide the facility to synchronize properly the operation of the decoder.

In practice, a decoder's free-running system clock frequency will not match the encoder's system clock frequency which is sampled and indicated in the SCR or PCR values. The decoder's System Time Clock (STC) can be made to slave its timing to the encoder using the received SCRs or PCRs. The prototypical method of slaving the decoder's clock to the received data stream is via a phase-locked loop (PLL). If however, a network varies the delay in delivering the data stream from the encoder to the decoder (as in the case of typical IP networks), such variations tend to cause a difference between the values of the time stamps and the values that they should have when they are actually received. This is referred to as SCR or PCR jitter. For example, if the delay in delivering one SCR is greater than the delay experienced by other similar fields in the same program, that SCR is late. Similarly, if the delay is less than for other clock reference fields in the program, the field is early. These timing jitters would result in variations in the frequency of the recovered clock. In applications where a significant amount of jitter is present at the decoder input and there are tight constraints on the frequency slew rate of the STC, the constraints of reasonable additional decoder buffer size and delay may not allow proper operation.

[1] adapted from ISO/IEC 13818-1 Annex D

7 Classification of Real-time Audiovisual Streaming Services

Depending on the characteristics of the real-time stream services, it may be possible to apply different schemes to overcome the jitter problem, for example a "pull" mechanism may be used in an retrieval environment while a "push" method has to be adopted in a broadcast scenario. It is therefore appropriate to classify the services according to the following categories.

7.1 Retrieval Services

This category includes services such as video-on-demand where a back channel is typically available and the source time base may be controllable.

7.2 Communicative Services

This category includes services such as video conferencing (which may use IP multicast) where the applications are sensitive to delay and response time, and the source time base may not be controllable.

7.3 Distributive Services

This category of services includes broadcast/multicast video, digital TV where the source time base may not be controllable.

8 Jitter Concealment Schemes

8.1 Frame Slipping Scheme[2]

In some applications where precise decoder timing is not required, the decoder's system time clock may not adjust its operating frequency to match the frequency represented by received SCRs (or PCRs). It may simply have a free running 27 MHz clock instead and depends on this decoder’s clock as the clock master. The pictures and audio samples are decoded at the time indicated by the decoder’s STC, which may be presented at slightly different times, according to the locally produced sample clocks. Depending on the relationship of the decoder’s sample clocks to the encoder’s system clock, pictures and audio samples may on occasion be presented more than one each or not at all (unless re-sampling is performed). This is referred to as “frame slipping” or “sample slipping”, in the case of audio. There may be perceptible artifacts introduced by this mechanism. Optionally, an adaptive re-sampling scheme may be used to avoid audio sample slippage.

The below diagrams illustrate the mechanism in the time domain for both cases where the server’s clock is slower and faster than the decoder’s clock respectively.

[2] This scheme may be apply to all service categories

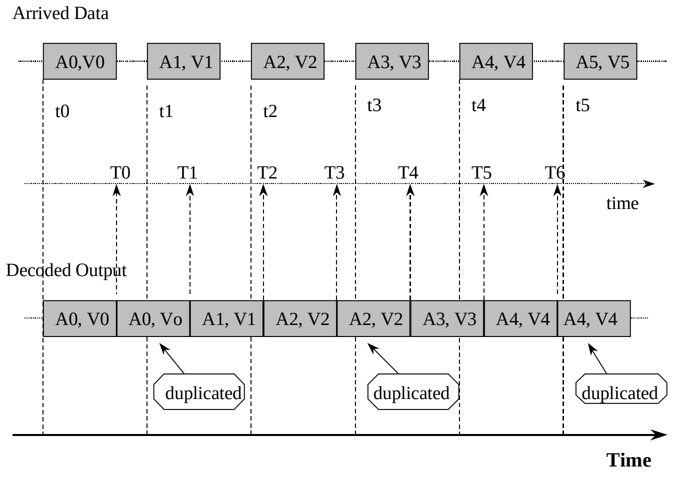

Figure 8-1 Frame Duplication – Decoder's Clock Positively Skew

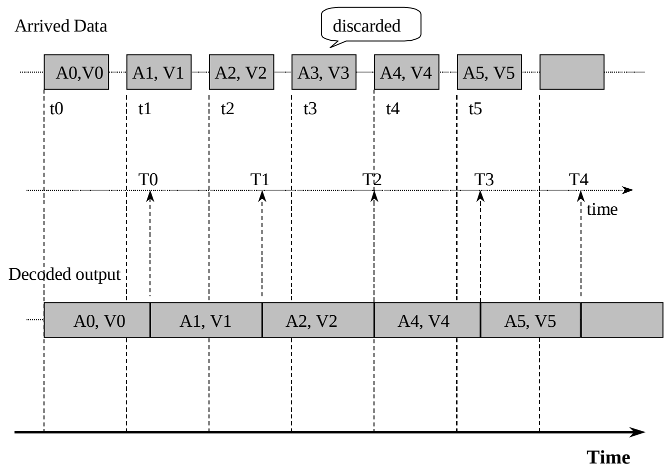

Figure 8-2 Frame Skipping – Decoder's Clock Negatively Skew

8.2 A "Pull" Scheme using Request-Reply[3] (RQRP)

Where applications can tolerate slight differences in the decoder's clock frequency from the encoder, but not the artifacts that may be produced as a result of repeated or deleted audio/video presentation units, a modified scheme based on Request/Reply (RQ/RP) may be adopted. Figure 8-3 illustrates such a buffer-based Request/Reply scheme.

[3] This "pull" scheme may be applied to applications classified under the retrieval services. Where applied within the DAVIC Intranet, the SDP <transport> sub-field of the 'm=' parameter should indicate <RQRP/RTP/AVT> (see SDP section of the DAVIC Intranet specification).

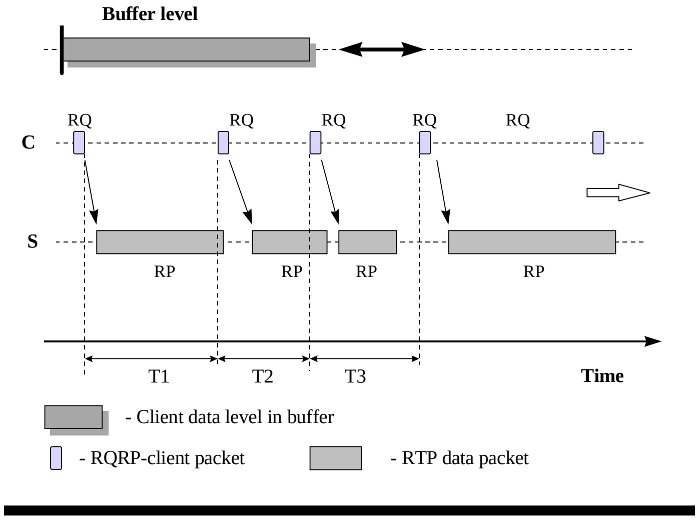

Figure 8-3 RQRP Operation for Buffer Based Scheme

Under the Request-Reply scheme, the client becomes the clock master. It assumes control of the session, and continuously issue stream requests to the Server, who then respond with the delivery of a designated number packets of MPEG streams. Provided that bandwidth and delay requirements has been satisfied, this scheme ensures that the stream delivery is always synchronized according to the decoder's system time clock (i.e. no early or late arrival of data) and hence no repeated/deleted frames or samples will be resulted. Through the use of the back channel, this scheme therefore ensures a high quality of MPEG playback without any frames or samples discarded or duplicated. It also reduces the effects of latency and jitters introduced by the network since the decoder is no longer dependent on the precise reception of MPEG packets to synchronize its time clock. The architecture of the decoder is also simplified, as there is no need for the use of phase-lock loop circuitry within the decoder.

A sample of the internal architecture of the client is shown in the diagram below. RTP is used to encapsulate the MPEG packets, while RTCP is used to estimate network traffic information including jitters and roundtrip delays. A buffer is used at the client to smooth out the effect of network jitters. The MPEG data received over the network is stored in this buffer and the client separately draws the data out at the required decoding rate and feeds the data to the MPEG decoder. At the same time, this buffer is also used to act as an indicator for determining when to issue requests to the server.

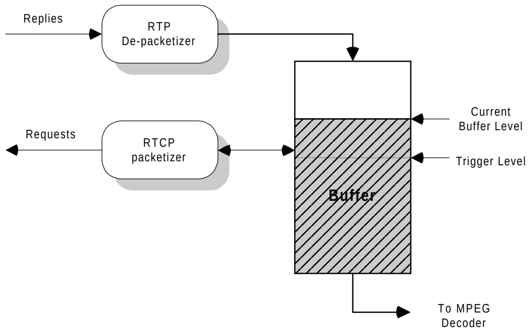

Figure 8-4 RQRP Client Architecture

The MPEG driver continuously reads from the buffer memory and trigger requests to the RTCP packetizer whenever the amount of data left in the buffer drops below certain pre-set trigger level. The RTCP packetizer then creates a RQRP Request packet specifying the amount of available buffer memory size, and transmit it to the video server. The video server, upon receiving the request, anticipates the amount of MPEG data that would have been consumed during the interval between the client issuing the request and receiving the reply, and sends back reply packet(s) containing an appropriate amount of MPEG data.

8.2.1 Single Request / Single Reply

A number of variations of the buffer-based request-reply schemes are possible, without needing significant modification to the Server. In its simplest form, the client issues a single request to the server and awaits the reply before issuing the next request (see Figure 8-5).

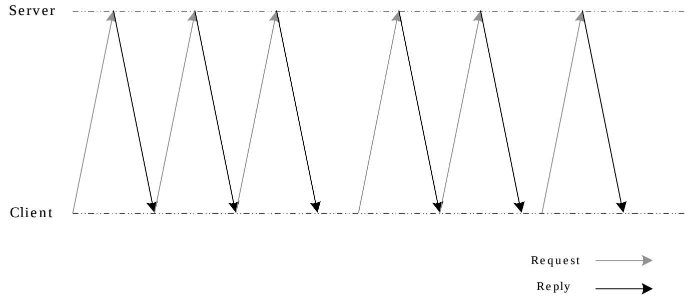

Figure 8-5 Single Request / Single Reply

Under this scheme however, the utilization of the network bandwidth is very low. In order to sustain the playback of the MPEG stream, the following constraint must be satisfied so that sufficient data is available while awaiting the next packet arrival,

Reply Packet Size >= (round-trip delay x MPEG bit rate)

8.2.2 Single Request / Multiple Replies

Instead of the Server delivering a single packet upon each request, the server could issue multiple packets. Furthermore, the number of replies may be variable depending on the availability of the client's buffer; the client could ask for many replies in one single request when its buffer is relatively empty, and conversely when it is full, it request for lesser number of packets.

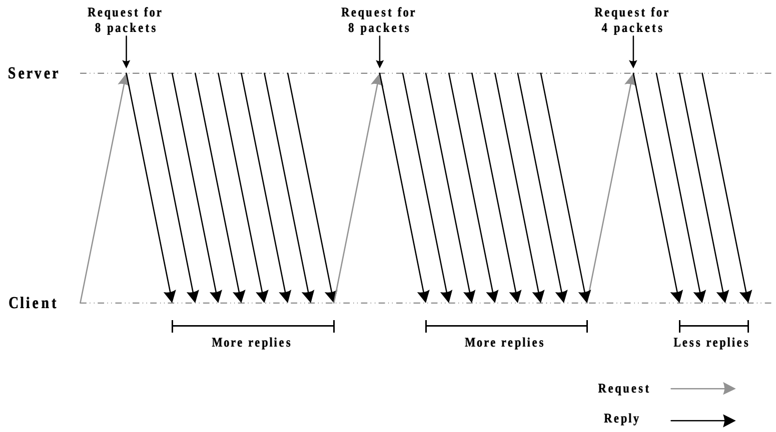

Figure 8-6 Single Request / Multiple Replies

The result of this is therefore a compromise of "push" and "pull" schemes such that the client's is still the clock master asking for packets when needed, but the server *pushes* more than 1 packet upon each request. This improves the network utilization since it results in overall more data being delivered upon each request overcoming the limit of RTP/UDP packet size.

8.2.3 Multiple Requests / Multiple Replies

While the above scheme improves the network utilization, it still incurs the waste of network resources during the roundtrip delay upon each request. Further improvements can be made by issuing requests in advance whenever there are any buffer space available on the client, without the need to wait for the arrival of replies from the previous request. Where the network is full duplex, this will certainly improves the network utilization. This scheme can also be combined with multiple replies, such that requests are sent whenever there are any buffer available, and the replies are also in multiple packets. The combined scheme is illustrated in Figure 8-7.

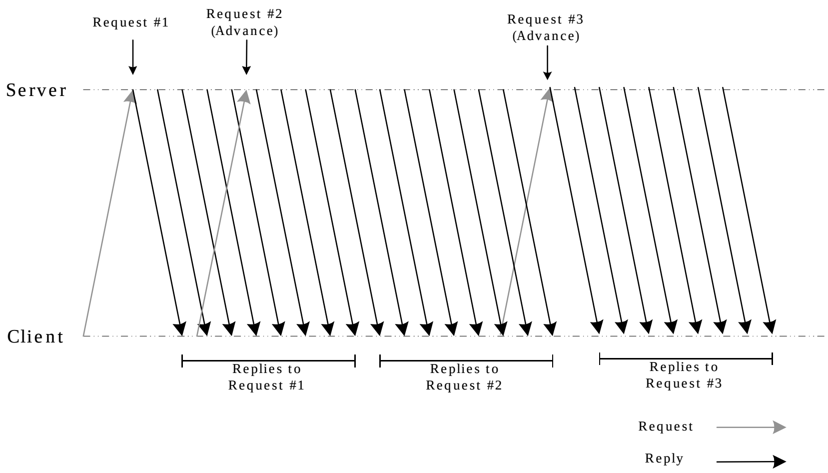

Figure 8-7 Multiple Requests / Multiple Replies

8.2.4 Congestion Control

While the use of multiple replies improves the network utilization, an inherent problem is the possibility of packet loss due to network congestion when the Server *pushes* the replies at a rate faster than the rate that the network could handle. This problem may be particularly significant during the start of the client’s operation where the client’s buffer is empty and issues many consecutive requests to the Server. A fast Server may then respond at a rate that is higher than the network or receiver capacity. To deal with this problem, some QoS mechanisms such as RSVP must be used to guarantee the network bandwidth. Furthermore, an algorithm adapted from TCP's ***slow start*** (Jacobson, 1988) may be used to adapt to fluctuating network traffic.

When a session is first established, the client initializes an internal variable - *Maximum replies* to 1 and then issues a request for only 1 reply packet. If this reply is acknowledged before the timer goes off, it doubles the *maximum replies* and requests for two replies. As each of these replies is acknowledged, the *maximum replies* are progressively increased. The *maximum replies* keeps growing exponentially until either a timeout occurs or the internal receiver's network buffer is reached. The idea is that if burst of size, say, 1024, 2048 and 4096 bytes work fine, but a burst of 8192 bytes gives a timeout, the maximum number of replies that the client should request should be set to 4096 to avoid congestion.

Related to the above, the retransmission timeout is also a potential issue. If the timeout is set too short, unnecessary retransmissions will occur, clogging the network with useless packets. If it is set too long, performance will suffer due to the long retransmission delay whenever a packet is lost. The solution is to use a highly dynamic algorithm that constantly adjusts the timeout interval, based on continuous measurements of network performance. For each connection, the client maintains a variable, *RTT*, that is the best current estimate of the round-trip time to the Server. This estimate may be derived from the round-trip time computation used in ***RTCP***. A suitable retransmission timeout may then be set as βRTT, where β may be an arbitrary value such as 2.

8.2.5 Incorporating RQRP into RTCP

RQRP may be incorporated into RTCP through extension of the APP message without modifying the RTCP packet structure. The RQRP Request message is defined as a subtype of the RTCP-APP packet. The ASCII text – "Buffer-based RQRP" is further used for identification purposes. The general message format of the proposed RQRP message is specified at below.

<table>
<tr><td>v=2</td><td>p</td><td>Subtype = RQRP</td><td>PT = 204</td><td colspan="2">Length</td></tr>
<tr><td colspan="6">SSRC/CSRC</td></tr>
<tr><td colspan="6">name = "BUFFER-BASED-RQRP"</td></tr>
<tr><td colspan="4">Sequence_number</td><td>Packet_size</td><td>Packet_count</td></tr>
</table>

Figure 8-8 RQRP Request Packet Format

Subtype: A field that defines a sub-type of the RTCP packet content type. In this case it is RQRP, indicating an extension of the header defining additional parameters for the RQRP message. The actual value is to be determined.

PT: RTCP packet type. In this case it is 204, indicating an APP packet.

Length: The size of the RTCP packet.

SSRC/CSRC: Synchronization source/Contributing source – send of RTP packets

name: An ASCII description of the RTCP packet content

Sequence_number: A 2 octet field containing the sequence number of the first RTP packet to be transferred.

Packet_size: a 1 octet field that conveys the size of RTP packet requested. The size is defined by the formula:

$$RTP\_packet\_size = 2^{(Packet\_size)}$$

Packet_count: The number of RTP packets to be delivered in response to this request.

8.3 A "Push" Scheme using "Half-Buffer based Software PLL"[4]

Where a "push" service (such as digital TV broadcast) is required, there is a large down-stream channel and small or no back-channel. This scheme employs RTP as the transport service and receiver buffering to achieve real-time AV playback. Jitter smoothing is achieved with the use of a FIFO buffer. A software PLL keeps the buffer approximately half full by adjusting the output rate in response to changes in buffer fullness. By monitoring the movement of the buffer filled position over a given period, the drift of the client's decoding clock compared to the server's encoding clock can be derived and adjusted. The following diagrams illustrate the conceptual operation of the scheme:

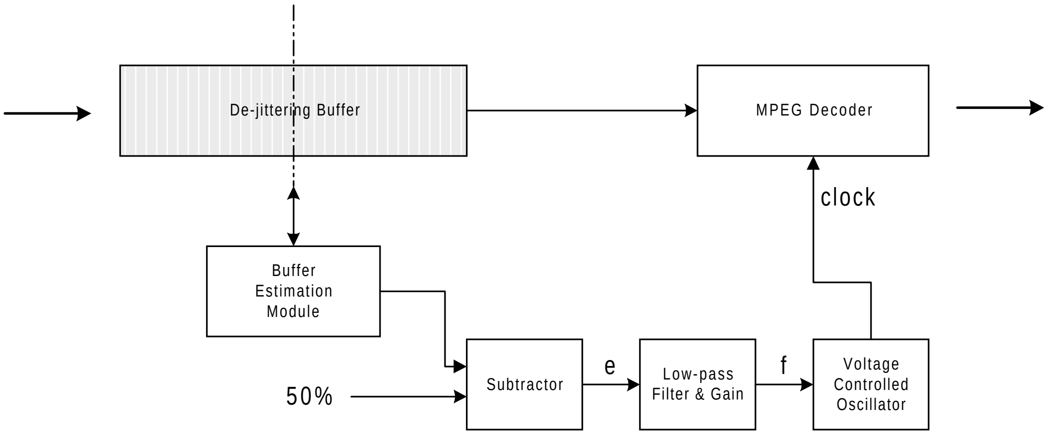

Figure 8-9 De-jittering and clock recovering operation

The *Buffer Estimation Module* continuously estimates the current buffer level and compares it with the intended half-buffer position. If the buffer level is less than 50%, a negative error signal is generated and sent to the *low-pass filter* and *voltage controlled oscillator* to decrease the clock frequency. Conversely, a positive signal is sent to decrease the frequency when the buffer level decreases.

A problem with the above scheme is the effect of network jitter which may potentially affect the estimation of the clock drift. In order to reduce this effect, the average buffer level L should be estimated and adapted over a wide range of values, and appropriate noise reduction and gain control should be introduced in order to reduce the effect of sudden changes in the buffer level as network packets arrive. An example of a possible method for computing the buffer level based on the instantaneous value M may be as follows:

$$L = \alpha L + (1 - \alpha) M$$

where α is a smoothing factor between 0 and 1 that determines how much weight is given to the old value.

Typically, a high α gives more weight to older values while a low α gives more weight to the new value.

[4] This scheme may be use in applications classified under the communicative and distributive services. It is adapted from ISO/IEC 13818-1 Annex D