Temporally Redundant Audio Format

Ashley Butterworth
Apple Inc

The Problem

- Some network mediums (such as WiFi) don’t provide the same level of packet delivery reliability as wired Ethernet does.
- In particular there are times when sequences of packets may be lost due to interference, traffic collisions, or other inaccessibility of the medium
- The streaming model that we use in 1722 doesn’t handle this well, and we typically don’t want to use the retransmission methods provided by the network medium.

A solution

- One solution to the problem is to send 2 copies of the stream with a time offset between the packets of the streams where the time offset between them is large enough to overcome the largest expected packet dropout period.
- This however requires 2 packetizers and 2 depacketizers and uses extra bandwidth for the extra headers.

- The proposal is to put both sets of samples in the same packet.

- This is not for ultra-low latency applications, the latency has to be at least large enough to cover the expected dropout period + max transit time

How it works

- Packet will look a lot like AAF with extra redundant audio payload data
  - If possible would like to make it “compatible” with a AAF listener by ignoring the extra data
- Frame Conversion Time (see Fig 6 1722-2016) contains the Max Allowed Dropout Time (MADT)
- Redundant audio data will have a “presentation time” that is offset from the stream presentation time by MADT (redundant data presentation time = presentation time + MADT)
  - This means that the redundant audio is delivered _before_ the primary audio
  - MADT is communicated out of band by 1722.1

Why redundant data is in the future

- Max Transit Time is already well defined by 1722-2016, and it’s a good definition!
- We can keep the Max Transit Time independent of the Max Allowed Drop Time

- By not changing the primary audio the packet could potentially be delivered to a well constructed AAF receiver and played back aligned with the redundant audio receiver

Packet Format

<table>
<tr><td></td><td></td><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td></td><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>subtype data</td><td>00</td><td colspan="8">subtype</td><td>sv</td><td colspan="3">version</td><td>mr</td><td colspan="2">rsv</td><td>tv</td><td colspan="8">sequence_num</td><td colspan="7">reserved</td><td>tu</td></tr>
<tr><td>Stream ID</td><td>04
08</td><td colspan="32">stream_id</td></tr>
<tr><td>AVTP Time</td><td>12</td><td colspan="32">avtp_timestamp</td></tr>
<tr><td>Format Specific</td><td>16</td><td colspan="8">format</td><td colspan="4">nsr</td><td colspan="2">rsv</td><td colspan="10">channels_per_frame</td><td colspan="8">bit_depth</td></tr>
<tr><td>Packet info</td><td>20</td><td colspan="16">stream_data_length (octets)</td><td colspan="3">rsv</td><td>sp</td><td colspan="4">evt</td><td colspan="8">reserved</td></tr>
<tr><td>Primary Audio payload</td><td>24</td><td colspan="32">primary_audio_data_payload</td></tr>
<tr><td>Redundant Audio payload</td><td>24+ primary_audio_data length</td><td colspan="32">redundant_audio_data_payload</td></tr>
</table>

Example

- 48kHz, packet every 125us (6 samples per packet), 10ms MADT
- Samples are numbered 0, 1, 2, 3, …
- First packet contains samples
  - primary_audio_data: 0, 1, 2, 3, 4, 5
  - redundant_audio_data: 480, 481, 482, 483, 484, 485
- Second packet contains samples
  - primary_audio_data: 6, 7, 8, 9, 10, 11
  - redundant_audio_data: 486, 487, 488, 489, 490, 491

Example

Continued...

- 99th packet contains samples
  - primary_audio_data: 480, 481, 482, 483, 484, 485
  - redundant_audio_data: 960, 961, 962, 963, 964, 965
- 100th packet contains samples
  - primary_audio_data: 486, 487, 488, 489, 490, 491
  - redundant_audio_data: 966, 967, 968, 969, 970, 971

Questions and comments