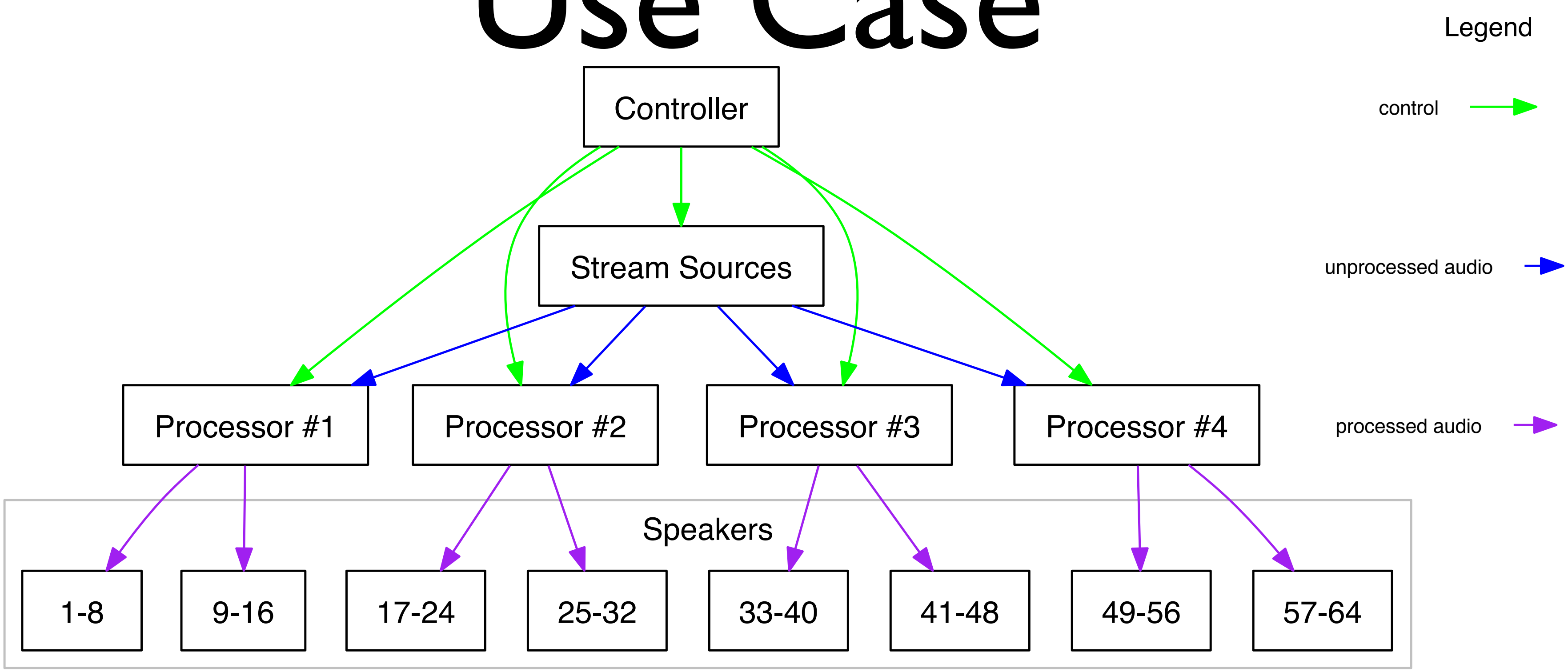
AVB Control Streams Pro Audio Use Cases

Jeff Koftinoff <jeffk@meyersound.com>
2011-11-22

Use Case



- Correlated audio is processed by 4 audio processors, each generating two 8 channel streams.
- Processed audio is output by 64 speakers
- Controller adjusts parameters on all processors which affect audio on all speakers simultaneously

Use Case

- Audio control points such as mute and level are applied in all processors in order to adjust the sound field for the audience.
- If the audio control points are not applied on the correct sample in all processors, the audience will perceive annoying discontinuities in the sound field.
- Because of processing audio buffering, a 10 μs delay in one packet could cause a much larger delay to apply the desired change.

Use Case

- Sample accurate control point changes are required
- Control point changes and audio streams are both effectively part of the live performance
- AVB audio during a live performance has bandwidth guarantees via SRP, and AVB Control Streams need the same guarantees for a simple lost or delayed 'unmute' can be disastrous during a show

Summary

- We are now using distributed systems with related media and control
- We expect AVB media streams to be low latency with accurate presentation times, using IEEE Std 802.1Q-2011 (FQTSS and SRP)
- Control points affecting audio must be applied at the correct time
- Especially important when control events relate to performer events and audio content events
- Control events on multiple devices with related media are very sensitive to the relative timing of control point changes
- We need real time controls to also benefit from the capabilities of AVB protocols - even if they are not IEEE P1722.1 based control protocols.

Control Streams

- IEEE Std1722-2011, with iec61883-6 AM824 already supports time division multiplexed MIDI messages which may contain control messages and MIDI System Exclusive messages with sub-microsecond accurate time stamps.
- While this combination of packet encodings can be used for the transport of time sensitive control streams, a more efficient and flexible time sensitive control stream format is desired.

1722a Control Streams

- A new IEEE 1722 subtype
- Multicast packet contains Stream ID and presentation time
- Transports TDM'd packet based and stream based control protocols
- Contain EUI64 to define encapsulated control protocol information (Control Format Descriptor)

Control Format Descriptor allows:

- Arbitrary IANA TCP/UDP protocols by IANA assigned port
- Existing real time protocols such as Flexray, MOST bus, CANbus, Open Sound Control
- Vendor specific control protocols using organization's OUI24 or OUI36

IEEE P1722a Control Streams

Desired to be usable as a transport for:

- Existing UDP and TCP control protocols.
- CAN bus, MOST bus, and Flexray networks for automotive control
- Open Sound Control (OSC)
- IEEE P1722.1

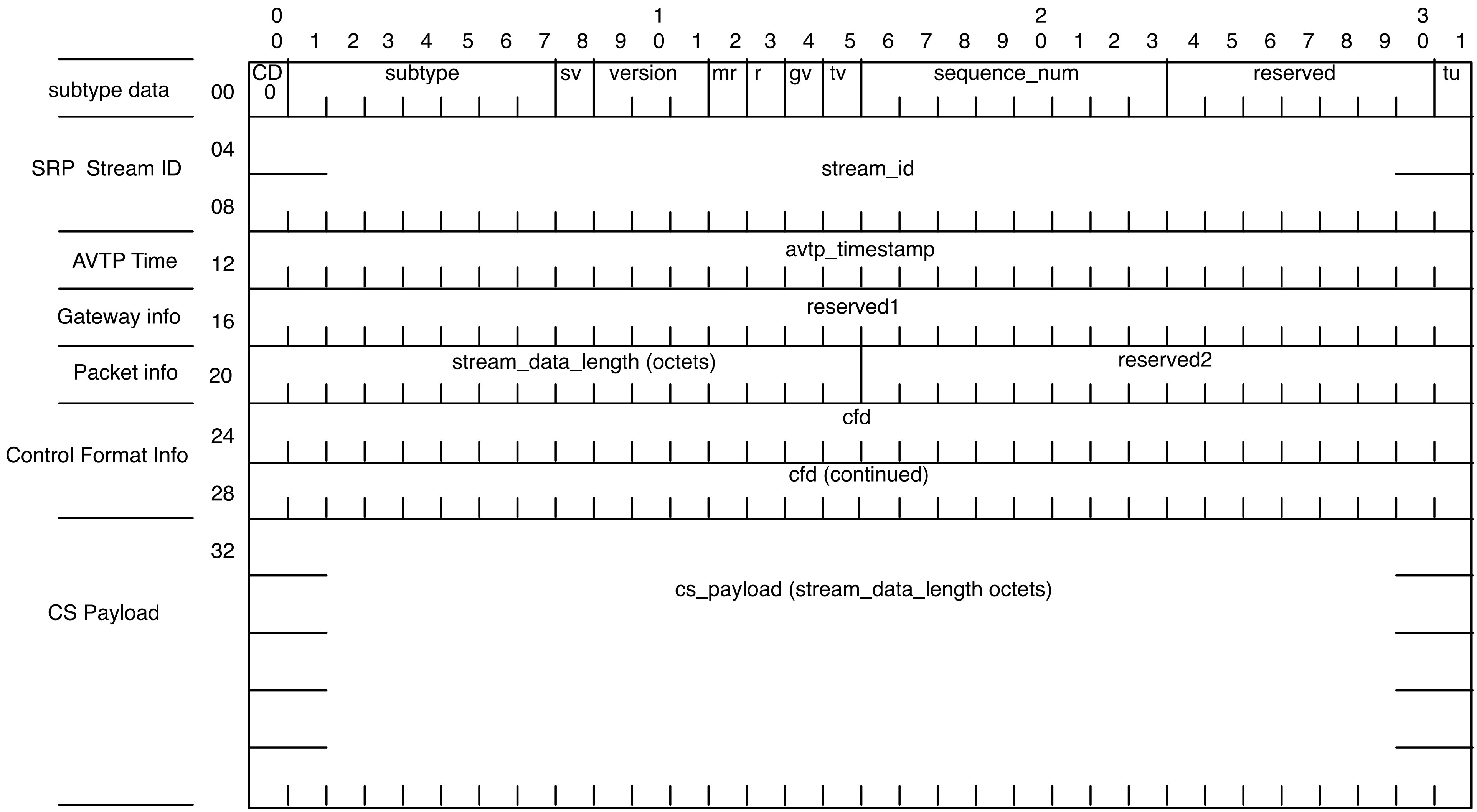
Controllers Need:

- Accurate time stamping
- Stream Reservation Protocol support
- Ability to source IEEE 1722a control streams
- To choose reasonable time in the future to use for presentation time
- Ability to sink IEEE 1722a control streams if any time sensitive response back is needed

Specifics

- IEEE P1722a Control Streams would be used for changing parameters
- Redundant AVB Control Streams would be used if high reliability is needed
- “Legacy” Layer 2 or Layer 3 protocols would be used for queries and large data transfers and metering
- A simplified AVTPDU can be used to allow customization headers for different CFD’s

Suggested AVTPDU



Suggested fields

- **CD** would be "0" (data)
- **subtype** would be 0x04
- **cfd** would be a EUI64
- **reserved1** and **reserved2** would be used as needed and defined for each **cfd**
- **cs_payload** would be the control data payload
- tv would be interpreted depending on **cfd**
- All other fields would be interpreted in the standard way as per IEEE 1722-2011

cfd definitions

- First 24 or 36 bits of the **cfd** field are IEEE assigned OUI24 or OUI36
- Any vendor can use their OUI in the cfd and use the reserved1 and reserved2 fields as they see fit
- cfd's with the OUI24 of "90-E0-F0" are for IEEE 1722 defined standard control protocols

90-E0-F0-00-00-00-YY-ZZ

- Specifies a tunneled IEEE 1722 control protocol
- YY specifies the **CD** and **subtype** fields of the control protocol
- ZZ specifies the **version** field of the control protocol

90-E0-F0-01-00-00-YY-ZZ

- Specifies a tunneled TCP socket protocol
- "YY-ZZ" specifies the IANA defined TCP port of the protocol

90-E0-F0-02-00-00-YY-ZZ

- Specifies a tunneled UDP socket protocol
- "YY-ZZ" specifies the IANA defined UDP port of the protocol
- Bit 14 of the reserved2 field specifies the "beginning of frame" (**bf**) flag, meaning the first octet of the **cs_payload** is the beginning of a UDP packet payload.
- Bit 15 of the reserved2 field specifies the "end of frame" (**ef**) flag, meaning the last octet of the **cs_payload** is the end of a UDP packet payload.
- The **tv** field is set if the **avtp_timestamp** field is valid.
- The **avtp_timestamp** field correlates with the **bf** flag.

Other standard cfd's?

- Flexray: 90-E0-F0-03-WW-XX-YY-ZZ
- Canbus: 90-E0-F0-04-WW-XX-YY-ZZ
- MOSTbus: 90-E0-F0-05-WW-XX-YY-ZZ
- Any others?
- How would these cfd's best utilize the **reserved1** and **reserved2** fields?
- Would they need TDM capabilities?