Potential problem with AAF and and MaxframeSize (MFS).

In general sample rates are asynchronous to PTP time. PTP provides a wall clock for time stamping but in AVB networks the PTP GM is typically running from a free running crystal with a 100PPM or 50PPM precision. It is quite likely that a media clock is received from another system outside the network which might be based on another free running 100 or 50PPM crystal or it could be derived from a GPS reference and be very close to nominal. We can't assume any relation between Media clocks and PTP. Likewise we cant assume any synchronous relation between the sample rate and the ethernet MAC clock.

Imagine an isochronous system providing exactly one MFS packet every observation interval as seen from that particular talker. Let's also assume that this end station has a crystal running somewhat faster so the actual observation interval of 125us is a bit shorter than the nominal 125us.
Now assume that this packet traverses a bridge with a crystal running a bit slower than nominal so the observation interval is a bit longer than the nominal 125us. In this case the bridge will actually see one packet and a tiny bit of a second packet in one observation interval.
That is fine due to the nature of the credit shaper. The shaper and the bridge does no actually observe the interval. The nature of the shaper shapes the traffic such that it complies with the interval. If the shaper was programmed based on the actual MFS + OH the shaper in the bridge would however push this 'early' frame out into the next observation interval from the point of view of its local clock.
802.1Q handles this situation by adding 1 byte which for a maximum frame (1500) is equivalent to an over subscription of 665PPM. If the MFS is smaller it is more. As all clocks in bridges and end stations are required to be within a limited PPM variation which assures less than 200PPM between any two clocks in the system. The added byte will always be able to accommodate for this variation in clocks.

In an audio system a Talker will be receiving samples at a rate which we call the sample rate or SR. A number of nominal sample rates exist such as 48kHz, 44.1kHz etc. Those are the nominal sample rates. For such rates various audio standards such as AES-11 and IEC60958-3 specify what the allowed deviation is. For professional audio the specification depends on the grade and vary from +/- 2PPM to +/- 50PPM. In consumer the variations are much larger.

In the original 1722 specification based on IEC61883-6 the Talker would look at how many samples it received in a given observation interval and send those. As an example if the device was receiving samples from an external source (SPDIF input) with a rate of 48k + 2PPM and the device was running on a clock with happened to be spot on nominal with an observation interval of 125us. In this case most packets would have 6 samples in them but for every 500,000 samples there would be 7. The IEC specifies that you should calculate the MFS based on 7 samples. While this would be overkill it would assure that the shaper would never push data out and eventually lose a sample.

When AAF was created the intention was to create a format which would use fixed size packets and reduce overhead. The effect was that for a nominal rate of 48k a Talker would always send 6 samples. In the example above this would mean that the Talker would send data 2PPM faster that what the MFS specified as it would be calculated based on the sample rate being 0PPM off the mark.
The reason why this would work through the switches is actually due to the extra byte and the fact that it over-reserves through the switches. However if the end station does not add 1 as well when setting operIdleSlope it would keep pushing the packet to the next nominal interval and eventually after 3,000,000 samples having pushed it out by a complete interval.

It is my opinion that this was wrong from the start. The extra byte was added to accommodate for the PPM variation in switches when operating the shaper and not to be able to send more information that was reserved. The group might have concluded that the few PPM extra (50 in pro) could be covered by the extra byte in the bridge specification. This could be fatal if a new TSN was created that found another way of accommodating for the bridge observation interval and one that would somehow strictly enforce the MFS.
In reality the AAF should have specified an overhead be added to the MFS instead of relying on the oversubscription of the underlying transport.

This also opens another can of worms. You are really only allowed to send MaxIntervalFrames per interval (the local device perception of an interval) but with AAF a small fraction more is sent as in the example above. With the current shaper that is okay as it has to accommodate for the aforementioned PPM variations in the switches but technically it is not compliant and could break if another shaper mechanism was implemented.

In ProAV there are other odd rates which a still considered as relating to a nominal rate. Like 48k pull-up which is nominally 48k pulled up by 1000PPM (48,048) to match video frames. Again adding up to 2 bytes for a maximum size packet would fix this with the current shaper but it actually breaks the rule of MaxIntervalFrames as it sends one thousands of a second frame within the same observation interval.
I have not yet red up on the new TSN specification but if we are using tricks like these we must assure that they are compatible with future shaper implementations.

Here are a list of typical rates which will be used in ProAV especially in relation to video:
44.056k
44.144k
44.1k
48.952k
48.048k
48k
Also their double and quadruple versions will exist.

I think this could be a long discussion but with the usual rates like 48, 96 and 192 maybe it is okay to keep things as they are and use the argument that the variation of the sample rate from nominal is less than the variation of the bridge clocks. A talker would still need to apply the extra byte to its local shaper in order not to push data out at the correct rate. This principle however would not work for the pull rates and it might be required to add an extra sample to each packet which does not always contain valid data like it is done with 44.1k in the current spec.

Lastly, there was a reason for IEC61883-6 to do it the way it did. Actually the way AAF does 44.1k is not that different from IEC but someone saw that 48k nicely divided by 8k not taking into account that those two are asynchronous.

AES67 has taking this further so the sample rate is derived from PTP alone so the first sample event of any rate was at epoc so the media clock can be generated from PTP time and nominal rate alone. That does mean that at 48k there will be exactly 6 samples per 125us PTP interval. At any pull rates it also means that at any interval in PTP time it is predictable exactly how many samples will be in a given interval. This method does have lot of drawbacks especially in systems which do not have a GPS based GM and you can't drive your media clock from an asynchronous source unless you drive the GM from that source. This is however less of a problem in broadcast where a global time reference is more or less guaranteed.
So while AES67 might have created a standard which can be problematic to deploy outside of broadcast, lets not do the same here and make one that can't be used in relation to broadcast.

Please add your comments. I think we need to look at this in two steps:
- Are we confident that the current Milan rates 48k, 96k and 192k are covered. Do we need to add a byte for the sake of the end station shaper?
- How are we going to support pull rates? What can we do to 1722 to make it work?

/Morten Lave

yes he is correct that the extra byte for reservation is there to allow the switch clock frequencies which measure the bandwidth to not fail.

Initially it was thought that this is also for media clock differences But later is was understood by the team that it is not. Obviously not captured properly.

With AAF the idea is that the talker sends a packet every time the packet is filled with samples. the traffic shaper shapes bits not packets.

Jeff Koftinoff 2019.02.01