One-Step, TCs, and AS-REV

Michael Johas Teener
mikejt@broadcom.com

Problem

- 1588 uses “transparent clocks” (TC) for best performance
  - peer-delay form can have equivalent performance with 802.1AS
    - but only if both 1588 and 802.1AS implementations are optimized
  - existing implementations are all one-step
    - that I’m aware off
- It would be nice to minimize differences between 1588 peer-delay TCs and 802.1AS TAS
  - meeting an 802.1AS TAS with one-step is the same as a 1588 peer-delay TC
    - at least for the “sync pipeline” that might be implemented in hardware

TC/TAS differences

- TC does not participate in BMCA, TAS does … TAS is basically a BC (boundary clock)
- TC alters ONLY the correction field and the MAC SA in sync messages
- TAS also changes sequenceId, sourcePortIdentity

BMCA?

- Frankly, all devices, even TC's should participate in BMCA, even minimally, otherwise management is difficult
    - Ask 1588 to consider making TCs participate in BMCA at a minimal level
- Regardless, BMCA processing is "control plane" and is not part of the synch pipeline

So, not really an issue

sequenceId

- TCs do not alter the sequence #
  - incremented only at the GM
  - TCs never add a sync
- TASs are spec’d like BCs
  - synthesize a sync on master ports if a synch is greatly delayed on the slave port (more than 30% over an expected sync interval)
  - need an independent sequence #

… but for one-step TAS

- Perhaps one-step master port is different:
  - sync transmit ASAP after sync receive
  - never synthesize a sync, just set a “late” flag for slave port for management
    - I note that we don’t already do that?
- Perhaps we just drop the sync timeout function completely?
  - Either follow received sequenceId (for “TC” master port), or synthesize (for “BC” master port)
  - Changes to portSyncSend actions

This is my preferred approach

sourcePortIdentity

- sourcePortIdentity is not really useful in a sync from a one-step TC/TAS except as an indicator of the source of sync time (BC or GM)
  - seems like its mainly useful for end-to-end delay processing, not used in 802.1AS

prefer that one-step 802.1AS master ports repeat sourcePortIdentity from slave port

Conclusion

- One-step 802.1AS TAS can act like 1588 P2P TC if both master and slave ports on the sync path are ***both*** one step
  - if slave port is two-step, TAS is as currently defined
  - if master port is two-step, TAS is as currently defined
- It's a straight-forward operation
  - I'll help Geoff get it in the draft
- There are NO requirements placed on systems that do not implement it

from January: “Legacy” compatibility

- One-step ***receive*** capability included in BMCA
- Use the twoStepFlag in the common header
  - If twoStepFlag is false in an announce message, then the port sending it can *receive* one-step sync
  - Current 802.1AS requires that twoStepFlag always be true, and ignored on reception

announce transmitter announce receiver	twoStepFlag set (only accept two step)	twoStepFlag clear (can receive one step)
two step only **(802.1AS-2011 or 802.1AS-REV two step only)**	ignored, will send back only two step	ignored, will send back only two step
one step rx OK **(802.1AS rev one step capable)**	accepted, will send back only two step	**accepted, will send back one step ONLY if capable**