1-step for 802.1AS Details
(v7, 9-September-2015)

Michael Johas Teener
mikejt@broadcom.com
Agenda

• Review of current proposal
• Media independent (Clause 10)
• Media dependent for Ethernet (Clause 11)
• Coordination with 1588

Note: I have an annotated PDF if you care to get the details
Review

• Most changes in Clause 11 (full-duplex point-to-point) media dependent layer
  • port can be “oneStepTransmit” (capable of one step Sync transmit) and “oneStepReceive” (capable of correctly receiving one step Sync) (per domain per port global variables)
  • if so capable, a master port can be in “oneStepTXOper” (per domain port global)
    • slave port operation updated to support 1step, but no mode change … already in AS-Rev/D1.0

• Intention is that media independent layer is almost the same
  • existing path unchanged
  • remove time filtering of Syncs in PortSyncSyncSend in “sync locked” mode
    • do not drop *early* Syncs or synthesizing late Syncs … that’s a media dependent thing
    • timeout still runs
  • use Signaling to communicate “oneStepReceive” capability and “syncLockDisable”
  • need to propagate some of the unprocessed received fields

• Notes and annex to explain implications of mixed 1step/2step networks
One major change

- **SourcePortIdentity** in Sync (and FollowUp) messages should be the portIdentity of the port on the current GM
  - **no change** to general messages (e.g., Announce)
- This was the original proposal for 802.1AS “way back when”
  - who knows why we changed it?
- The current meaning of **SourcePortIdentity** has no useful purpose in Sync/FollowUp
  - does it?
  - does *anybody* have a use for the current meaning?
Problem with sync interval?

• It’s really not clear to me what’s really supposed to happen when different ports have different currentLogSyncInterval values …

• it seems like we are trying to combine two separate modes into one state machine
  • one where a master port transmits a Sync as soon as possible after a Sync is processed by the slave port, and
  • and another where the master port has its own timing

• it’s pretty ugly
  • I’m proposing the outline of a solution, and I’ll help Geoff with the details
  • “syncLock” mode
• Clause 10 state machines unchanged except
  • master ports (sending sync) get some extra information propagated from the slave port (receiving sync)
    • propagate the received sourcePortIdentity and sequenceld
  • Add TLV to Signaling message to communicate “oneStepReceive” capability

• Clause 10 data structures:
  • Separate PortSyncSend and PortSyncReceive
  • MDSync<x> and PortSync<x> have new fields
    • rxSequenceld - holds sequenceld from the current slave port
  • <y>SyncSend have new fields:
    • rxSourcePortIdentity - received on the current slave port (maybe “GMPortIdentity” to be clear?)
  <x> is “Send” or “Receive”, <y> is “MD” or “Port”
Move info TLV to sync

• 802.1AS uses the Follow_Up to carry useful information
  • move it to the Sync if oneStepOperation is true

Table 11-10—Follow_Up information TLV

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>Octets</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>28</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Octets</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>tlvType</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>lengthField</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>organizationId</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>organizationSubType</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>cumulativeScaledRateOffset</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>gmTimeBaseIndicator</td>
<td>2</td>
<td>14</td>
</tr>
<tr>
<td>lastGmPhaseChange</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>scaledLastGmFreqChange</td>
<td>4</td>
<td>28</td>
</tr>
</tbody>
</table>
MDSyncReceiveSM

• No changes for one step except:
  • populating the MDSyncReceive structure from the Sync event message
  • including the TLV
  • include the sequenceld value
**MDSendSynchSM**

(“syncLocked”) FALSE

- Use **currentLogSyncInterval** for timing
  - Slave port may be different sync rate, so we need to synthesize the Sync

- Build sync from **MDSendSync** structure
  - uses the newest upstreamTxTime and egress timestamp (and other latency info) to add the residence time to the correction field (same as before)
    - perhaps use improved Noseworthy algorithm
  - uses locally generated sequenceId
  - uses received sourcePortIdentity for sourcePortIdentity
• Slave port must follow Master port Sync timing

• Build sync from MDSendSync structure
  • uses the upstreamTxTime and egress timestamp (and other latency info) to add the residence time to the correction field (same as AS-2011)
  • uses rxSequenceld for the sequenceld
  • uses received sourcePortIdentity for sourcePortIdentity
Two-step pDelay

• pDelay is infrequent
  • 1 per second, NOT duplicated for domains (or at least it shouldn’t be)
  • low processing load

• pDelay is NOT relayed
  • processing is local anyway, hardly anything to be gained with one-step
• Signaling is currently used to communicate with peer (per domain) port
  • rate for pDelay, sync and announce, computation of neighbor rate ratio, etc. in message
    interval request TLV

• Add one-step receive capability in the Signaling message
  • Hmm. I notice that we never define when Signaling messages are sent.
  • I also note that sometimes it’s “Signaling” and sometimes it’s “Signalling”
  • Should be sent about the same time pDelay process starts

• New TLV in message interval request Signaling message
  • new bit in TLV is “flags.oneStepReceive”
  • If oneStepReceive is TRUE, then the (per domain) port sending it can receive one-step sync

<table>
<thead>
<tr>
<th>signaling transmitter</th>
<th>oneStepReceive false (only accept two step)</th>
<th>oneStepReceive true (can receive one step)</th>
</tr>
</thead>
<tbody>
<tr>
<td>signaling receiver</td>
<td></td>
<td></td>
</tr>
<tr>
<td>oneStepTransmit false</td>
<td>ignored, will send back only two step</td>
<td>ignored, will send back only two step</td>
</tr>
<tr>
<td>(802.1AS-2011 or 802.1AS-REV two step only)</td>
<td>oneStepTXOper = false</td>
<td>oneStepTXOper = false</td>
</tr>
<tr>
<td>oneStepTransmit true</td>
<td>accepted, will send back only two step</td>
<td>accepted, will send back one step</td>
</tr>
<tr>
<td>(802.1AS rev one step capable)</td>
<td>oneStepTXOper = false</td>
<td>oneStepTXOper = TRUE</td>
</tr>
</tbody>
</table>
Notes on hybrid operation

• “Hybrid operation” means the path back to the GM includes both one step and two step links.

• There are two fields in sync/follow-up that now have possibly different meanings:
  • sequencelId
  • sourcePortIdentity on Sync / FollowUp
sequenceID

• as far as I can tell, sequenceID is not relevant end-to-end, it’s just a link parameter
  • only used to correlate sync with corresponding follow-up
• for a “syncLocked” path through a TAS, sequenceID is repeated …
  • never tested or validated
  • consider the case of transition from TC-like to non-TC-like and vice-versa
  • but always incrementing at nearest upstream TAS (non-syncLocked path) or GM
• perhaps just require that each sequenceID is different than the previous “n” sync messages
  • where “n” is TBD, maybe 4
sourcePortIdentity

• in 1588 the sourcePortIdentity is the identifier of the closest upstream GM or BC
• for 802.1AS-2011, all TAS’s are BCs
• for 802.1AS-rev, I propose that a TAS is NOT a BC
  • meaning that sourcePortIdentity is just like the 1588 meaning
  • it could be “grand master identity”
  • I think this would be really useful!
  • but that would be breaking 1588, but perhaps if we ask 1588 to allow profiles to make this change
1588 implications

- 1588 TCs don’t necessarily wait for a follow up on a master port
  - so they don’t/can’t convert a two step sync into a one step synch
- 1588 has no concept of different ports in a TC doing different things
  - like one step and two step in the same device
  - but then they don’t define ethernet and wifi ports, either
- Port capabilities in announce or signaling messages?
  - help their plug-and-play, they were thinking about things like this for profile interoperation
- Requests to 1588
  - Allow these features to be included in a profile spec
  - Perhaps part of the new layered architecture
All done!

Well, not exactly … tbd includes:
- state machine updates for “syncLocked”
- state machine updates for setting “oneStepTXOper” (copy and paste from 10.3.14)

### Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>2015-04-07</td>
<td>Initial version, TSN call 2015-04-08</td>
</tr>
<tr>
<td>v2</td>
<td>2015-04-08</td>
<td>Separate out “TC” mode, fix names, agenda</td>
</tr>
<tr>
<td>v3</td>
<td>2015-05-03</td>
<td>Updates to for final discussion on AS call 2015-05-04</td>
</tr>
<tr>
<td>v4</td>
<td>2015-05-07</td>
<td>Effect of not computing cumulative rate ratio, requirements for that, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>note about loss of accuracy if rate ratio not used</td>
</tr>
<tr>
<td>v5</td>
<td>2015-05-19</td>
<td>Residence time correction</td>
</tr>
<tr>
<td>v6</td>
<td>2015-09-07</td>
<td>Additional notes, change names to be consistent, using signaling for port</td>
</tr>
<tr>
<td></td>
<td></td>
<td>capabilities</td>
</tr>
<tr>
<td>v7</td>
<td>2015-09-09</td>
<td>More consistent with 802.1AS-rev/D1.0</td>
</tr>
<tr>
<td>v7</td>
<td>2015-09-09</td>
<td>Automatic syncLock, per domain oneStepTXOper</td>
</tr>
</tbody>
</table>