1-step for 802.1AS Details
(v3, 3-May-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
Most changes in Clause 11 (full-duplex point-to-point) media dependent layer

- port can be “oneStepCapable” (per port global variable array)
- if so capable, a master port can be in “oneStepOperation” (port global)
  - slave port operation updated to support 1step, but no mode change, just a “oneStepSync” flag set in the MDSyncReceive data.
- a master port in “oneStepOperation” can operate like a TC
  - meaning not updating some sync fields
  - which can happen only if the current slave port is in “oneStepOperation” and the sync rate is the same
  - not required … it’s OK if a master port does update all fields

Intention is that media independent layer is almost the same

- existing path unchanged
- BMCA is a bit different to communicate “oneStepCapable”
- need to propagate some of the unprocessed received fields

Notes and annex to explain implications of mixed 1step/2step networks
Media independent

• Clause 10 state machines unchanged except
  • master ports (sending sync) get some extra information propagated from the slave port (receiving sync)
  • Add TLV to Signalling message to communicate “oneStepCapable”

• Clause 10 data structures:
  • Separate PortSyncSend and PortSyncReceive
  • MDSync<x> and PortSync<x> have new fields
    • rxSequenceld - holds sequenceld from the current slave port
    • oneStepSync - set if slave port received a one step sync
  • <y>SyncSend have new fields:
    • rxRateRatio - the rateRatio received on the current slave port
    • rxSourcePortIdentity - the sourcePortIdentity received on the current slave port
  <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

<table>
<thead>
<tr>
<th>Bits</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
  - set the oneStepSync flag if appropriate
  - include the sequenceId value
• If port is operating in one step mode:
  • if oneStepOperation[rcvdPSSyncPtr] &&
    (syncInterval[rcvdPSSyncPtr] == syncInterval[txPSSyncPtr]) &&
    TCOperation[[txPSSyncPtr]] then “transmit like a TC”
    • we know the slave port is one step and using the same synch rate, so we can
      operate like a 1588 transparent clock
    • not required, unless the group decides that it should be
  • if !!oneStepOperation[rcvdPSSyncPtr] ||
    (syncInterval[rcvdPSSyncPtr] != syncInterval[txPSSyncPtr]) ||
    !!TCOperation[[txPSSyncPtr]] then “one step, not TC”
    • we operate just like a 802.1AS port except we send a one step synch (one step
      master)
    • details follow

• If a port is not operating in one step mode:
  • no changes from 802.1AS-2011
**MDSendSyncSM**

(“transmit like a TC”)

- **Build sync from MDSendSync structure**
  - uses the upstreamTxTime and egress timestamp (and other latency info) to add the residence time to the correction field
  - uses rxSequenceld for the sequenceld
  - uses rxSourcePortIdentity for sourcePortIdentity
  - uses rxRateRatio for cumulativeScaledRateOffset

- **Requirement that send synch happen “as soon as possible after receive synch”**
  - not certain how this can be specified
MDSendSynchSM
(“one step, not TC”)

• Slave port is two step or different sync rate, so we need to synthesize the one step event message
  • or we just don’t want TC-like operation
• 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 before)
  • uses sequenceld for the sequenceld
  • uses sourcePortIdentity for sourcePortIdentity
  • uses rateRatio for cumulativeScaledRateOffset
• In other words, same values as used in two step
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
“Legacy” compatibility

- One-step receive capability included 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”

- Use new TLV in announce message
  - one field in TLV is “1stepCapable”
  - If 1stepCapable is true in an announce message, then the port sending it can receive one-step sync

<table>
<thead>
<tr>
<th>announce transmitter</th>
<th>1stepFlag false (only accept two step)</th>
<th>1stepCapable true (can receive one step)</th>
</tr>
</thead>
<tbody>
<tr>
<td>two step only</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>$1\text{stepOperation} = false$</td>
<td>$1\text{stepOperation} = false$</td>
</tr>
<tr>
<td>one step rx OK</td>
<td>accepted, will send back only two step</td>
<td>accepted, will send back one step ONLY if capable</td>
</tr>
<tr>
<td>(802.1AS rev one step capable)</td>
<td>$1\text{stepOperation} = false$</td>
<td>$1\text{stepOperation} = true$</td>
</tr>
</tbody>
</table>
“Hybrid operation” means the path back to the GM includes both TC-like and two step links.

There are three fields in sync/follow-up that now have possibly different meanings:

- sequenceld
- sourcePortIdentity
- cumulativeScaledRateOffset
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 “TC path” through a TAS, sequenceID is repeated ...
  - Never tested or validated
  - But always incrementing at nearest upstream TAS (non-TC path) or GM
cumulativeScaledRateOffset

• for a “TC path” the cumulative rate ratio is unchanged ... the residence time is so short, this should not be significant
  • I’ve asked Geoff to do some simulations to test this hypothesis
sourcePortIdentity

- 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 “TC path” is NOT a BC
  - meaning that sourcePortIdentity is just like the 1588 meaning
- it’s possible we could redefine sourcePortIdentity for 802.1AS-rev
  - it could be “grand master identity”
  - but that would be breaking 1588, perhaps
1588 implications

• 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

• We will have to go to them with this idea as part of their new layered structure
  • they might actually like the idea
All done!

document history

<table>
<thead>
<tr>
<th>v1</th>
<th>2015-04-07</th>
<th>initial version, TSN call 2015-04-08</th>
</tr>
</thead>
<tbody>
<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>
</tbody>
</table>