Initial Description of Possible Use of Sync and FollowUp Messages with Peer-to-Peer Transparent Clocks in AVB

Geoffrey M. Garner (Consultant)
Gmgarner@comcast.net

SAMSUNG Electronics

IEEE 802.1 A/V Bridging TG 2006 Denver
2006.03.06

Outline

- Introduction
- Review of message exchange semantics for Boundary Clocks (BCs)/Ordinary Clocks (OCs) in IEEE 1588 Version 1
- Current proposal for FollowUp Peer-to-Peer (P2P) Transparent Clocks (TCs) for IEEE 1588 Version 2
- Possible Use of Sync and FollowUp with P2P TCs for AVB
- References

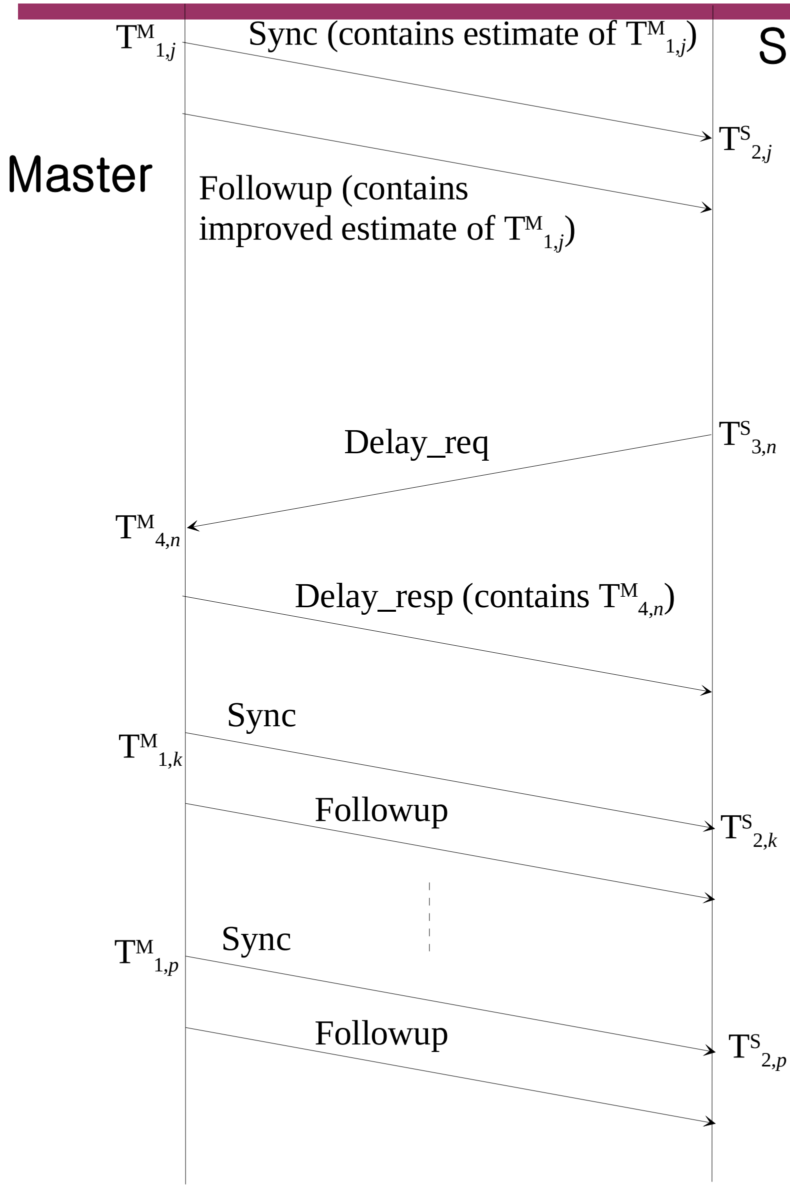
Introduction

- During the February 21, 2006 IEEE 1588/802.1 AVB design meeting [3], a way of using Sync and FollowUp with FollowUp Peer-to-Peer (P2P) Transparent Clocks (TCs) in AVB was discussed
- AVB will likely need a relatively short sync interval (e.g., 10 ms or less)
- AVB will use an inexpensive processor (e.g., 8051)
  - Consequence of this is that FollowUp messages may take up to 10 ms to process
- In conventional use of Sync and FollowUp, where Sync message is sent by Boundary Clock (BC) and traverses the chain of FollowUp P2P TCs, and FollowUp is sent by BC almost immediately after
  - Sync message is timestamped on entering and leaving P2P TC; residence time is TC is very short (i.e., << 10 ms)
  - FollowUp message is processed by TC; this takes on the order of 10 ms
  - Result is that a P2P TC that is a number of hops downstream will have received a number of successive Sync messages before it receives the FollowUp message corresponding to the first Sync message
  - This was felt to be undesirable (would require maintaining state information in TC)

Introduction (Cont.)

- The above could be alleviated in several ways
  - Increase the sync interval to be on the order of the number of hops multiplied by the FollowUp processing time (e.g., 70 ms for 7 hops) or greater
  - Process the FollowUp messages faster (i.e., use a faster processor)
- Both of the above were also considered undesirable
- An alternative method of using Sync and FollowUp was then discussed
  - This method avoids having multiple outstanding Sync messages when FollowUp arrives, without using the above two approaches
- The method was discussed verbally, and then documented in [5]
- The method is described here, after first providing background on message semantics for BCs and OCs in IEEE 1588 Version 1 and on the current P2P TC

Review of BC/OC Message Semantics in IEEE 1588 Version 1



Master−to−Slave Delay Estimate =
Sync_receipt_time – preciseOriginTimestamp

$$= T^S_{2,j} - T^M_{1,j}$$

Slave−to−Master Delay Estimate =
delayReceiptTimestamp – delay_req_sending_time

$$= T^S_{4,n} - T^M_{3,n}$$

one_way_delay (Propagation Delay Estimate)

$$\delta_n = \frac{\left(T^S_{2,j} - T^M_{1,j}\right) + \left(T^M_{4,n} - T^S_{3,n}\right)}{2}$$

Clock Delta (Offset from Master) =
Sync_receipt_time – preciseOriginTimestamp
− one_way_delay

$$= T^S_{2,k} - T^M_{1,k} - \delta_n$$

Review of BC/OC Message Semantics in IEEE 1588 Version 1

- Use current one_way_delay (propagation delay estimate) in in all computations of offset from master (clock delta), until new one_way_delay is computed

- This approach is mathematically equivalent to pure 2-way approach in white paper

Current Proposal for FollowUp P2P TC for IEEE 1588 Version 2 [1]

- Master sends Sync to slave (with followup flag set) and timestamps the Sync as it leaves
- Master send FollowUp to slave with timestamp (preciseOriginTimestamp)
- Each P2P TC timestamps the Sync as it enters and leaves, and computes the residence time
- Each P2P TC adds the residence time to the correction field of the FollowUp message that corresponds to the Sync message
- Each P2P TC adds the delay on the upstream link (i.e., the link on which the Sync message arrived) to the correction field of the corresponding FollowUp message
  - Link delays are calculated using the ADelay mechanism (see next 3 slides)
- Each P2P TC may use the preciseOriginTimestamp to adjust its frequency
  - E.g., P2P TC can measure the elapsed time of its local free-running oscillator between the first and last of a series of N Sync messages
    - P2P TC would obtain the elapsed time relative to the Master from the preciseOriginTimestamps corresponding to those Sync messages
    - P2P TC would compute its frequency offset relative to the slave, and use this frequency offset either to directly adjust its frequency or synthesize a corrected frequency (flextimer)

Current Proposal for Followup P2P TC for IEEE 1588 Version 2 [1]

- Measurement of link delay using ADelay mechanism
  - For now, we assume BC/OC nodes in AVB will use ADelay
    - IEEE 1588 Version 2 will not require a BC or OC to use ADelay (for backward compatibility with Version 1 equipment)
- Each P2P TC port measures the delay on the link attached to that port, independent of any master/slave relation
  - Therefore, the link delay is known to both endpoint nodes (P2P TC or BC/OC)
- The mechanism described on the next two slides is a specialized version of the mechanism in Section 7 of [1]
  - No End-to-End (E2E) TCs present
  - All P2P TCs are Followup TCs

Current Proposal for Followup P2P TC for IEEE 1588 Version 2 [1]

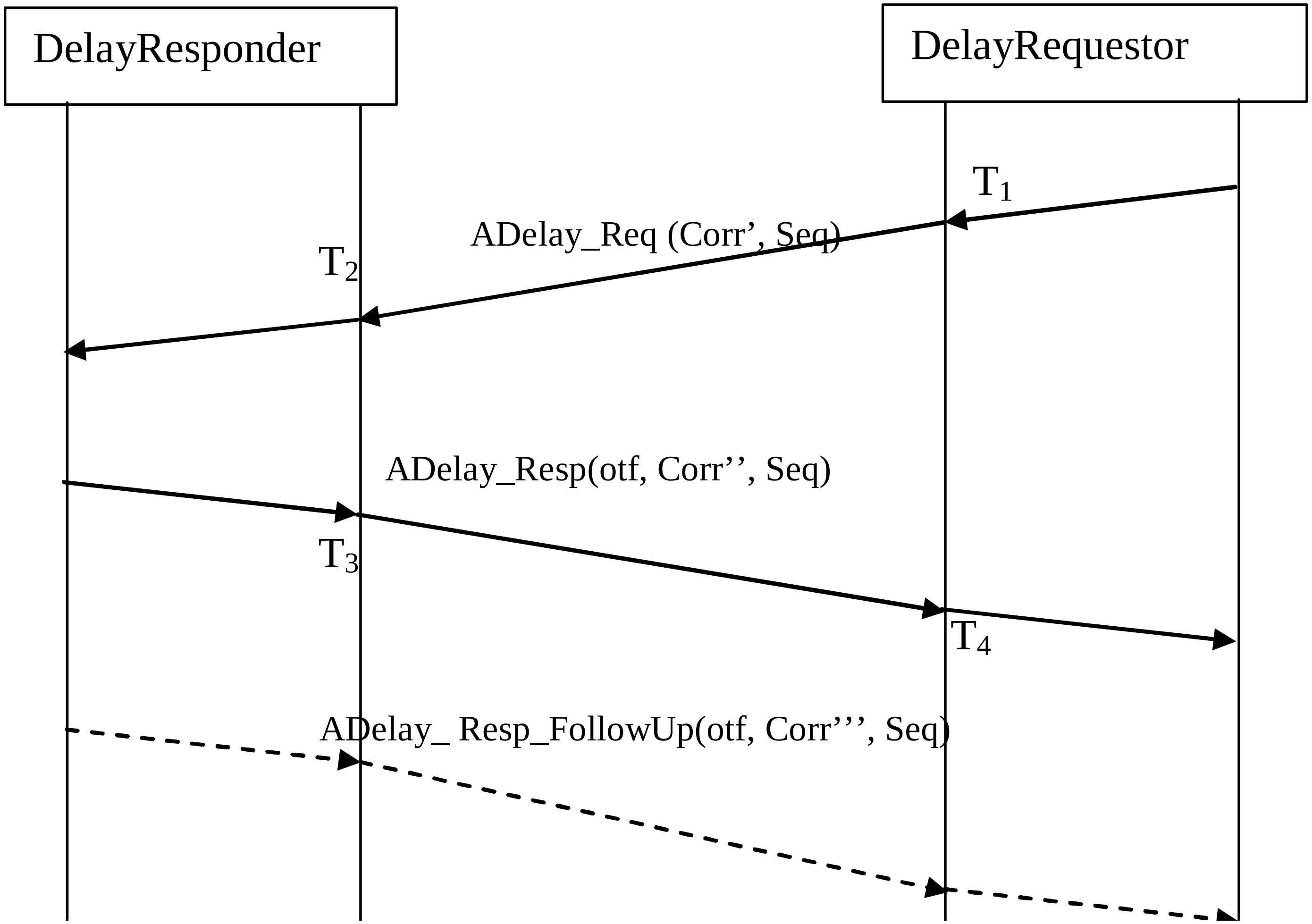


Figure taken from Section 7 of [1]

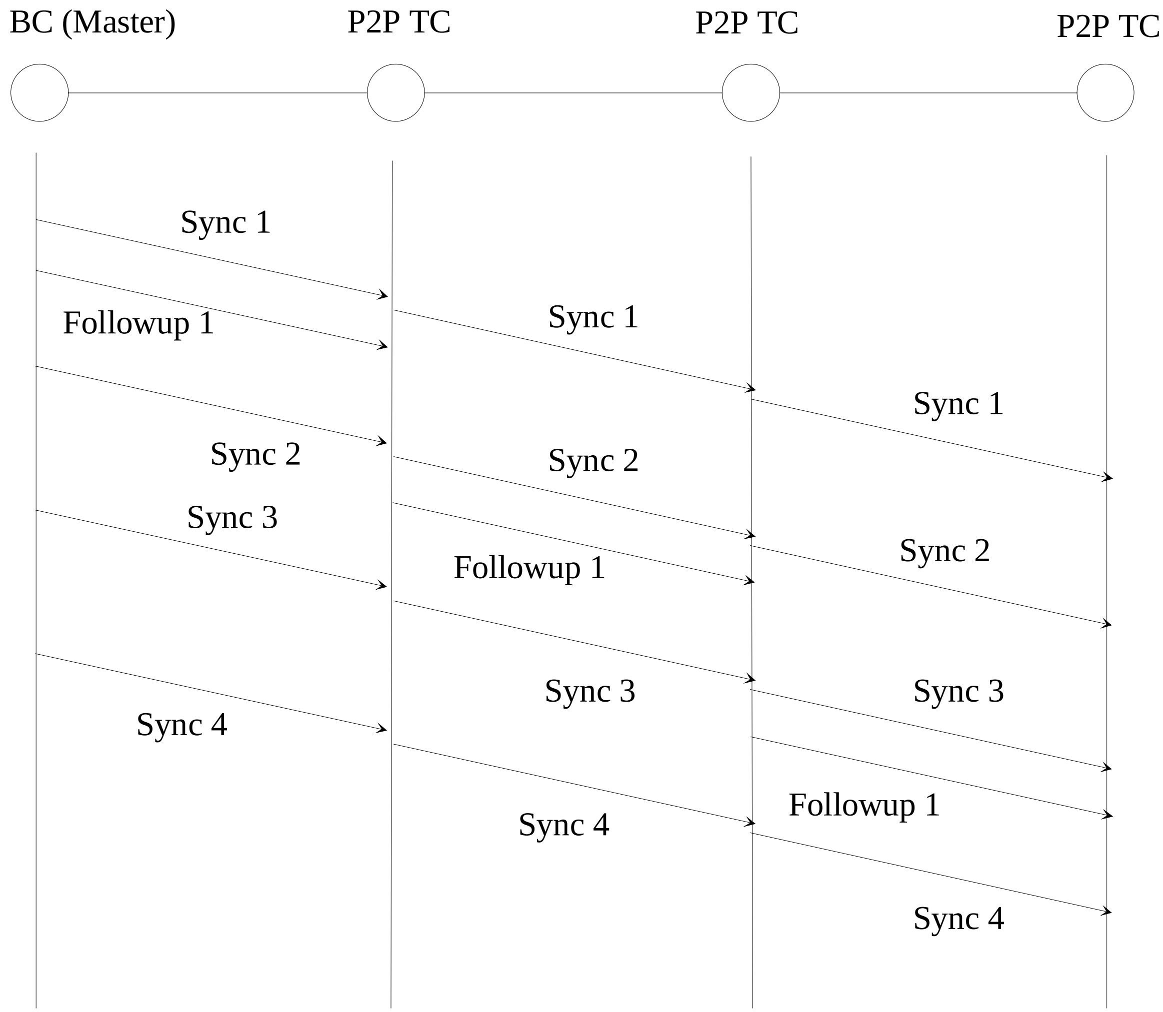
Specialization of Current proposal for Followup P2P TC for IEEE 1588 Version 2 [1]

- The end of the link that desires the path delay, i.e., the DelayRequestor, sends ADelay_Req to the other end (the DelayResponder)
- The DelayRequestor timestamps the ADelay_Req message as it is sent (timestamp value = T_1)
- The DelayResponder timestamps the ADelay_Req message as it arrives (timestamp value = T_2)
- The DelayResponder sends an ADelay_Resp message back to the DelayRequestor, and timestamps the message as it is sent (timestamp value = T_3)
- Note that T_1, T_2, and T_3 are not placed in the ADelay_Req or ADelay_Resp messages
- The DelayResponder computes the turnaround time $T_3 - T_2$ and places it in the correction field of the ADelay_Resp_FollowUp message
- The DelayRequestor receives and timestamps the ADelay_Resp message (timestamp value = T_4)
- The DelayRequestor receives the ADelay_Resp_FollowUp message and computes the link delay as $(T_4 - T_1 - \text{correction\_field})/2$

Use of Sync and FollowUp with P2P TCs for AVB

- A/V Bridges will use an inexpensive processor, e.g., 8051
- As a result, P2P TCs may require as much as 10 ms to process FollowUp message
- For a 7-hop path, FollowUp message may require 70 ms or more to travel from master to slave
- Sync message is forwarded quickly by P2P TC, and travels from master to slave in a time on the order of the AVB latency requirement or less (latency requirements in the range of 2 – 6 ms have been discussed)
- This means that when FollowUp corresponding to a particular Sync arrives at the slave, some number of additional Sync messages (e.g., 6 in this example) will have arrived
- The same will be true for the intermediate P2P TCs, although there will be fewer outstanding Sync messages for TCs closer to the master (illustrated on next slide for 3 hops)
  - In any case, P2P TCs and slave will have to maintain state information for multiple outstanding Sync messages
  - This is undesirable for an inexpensive processor

Use of Sync and FollowUp with P2P TCs for AVB

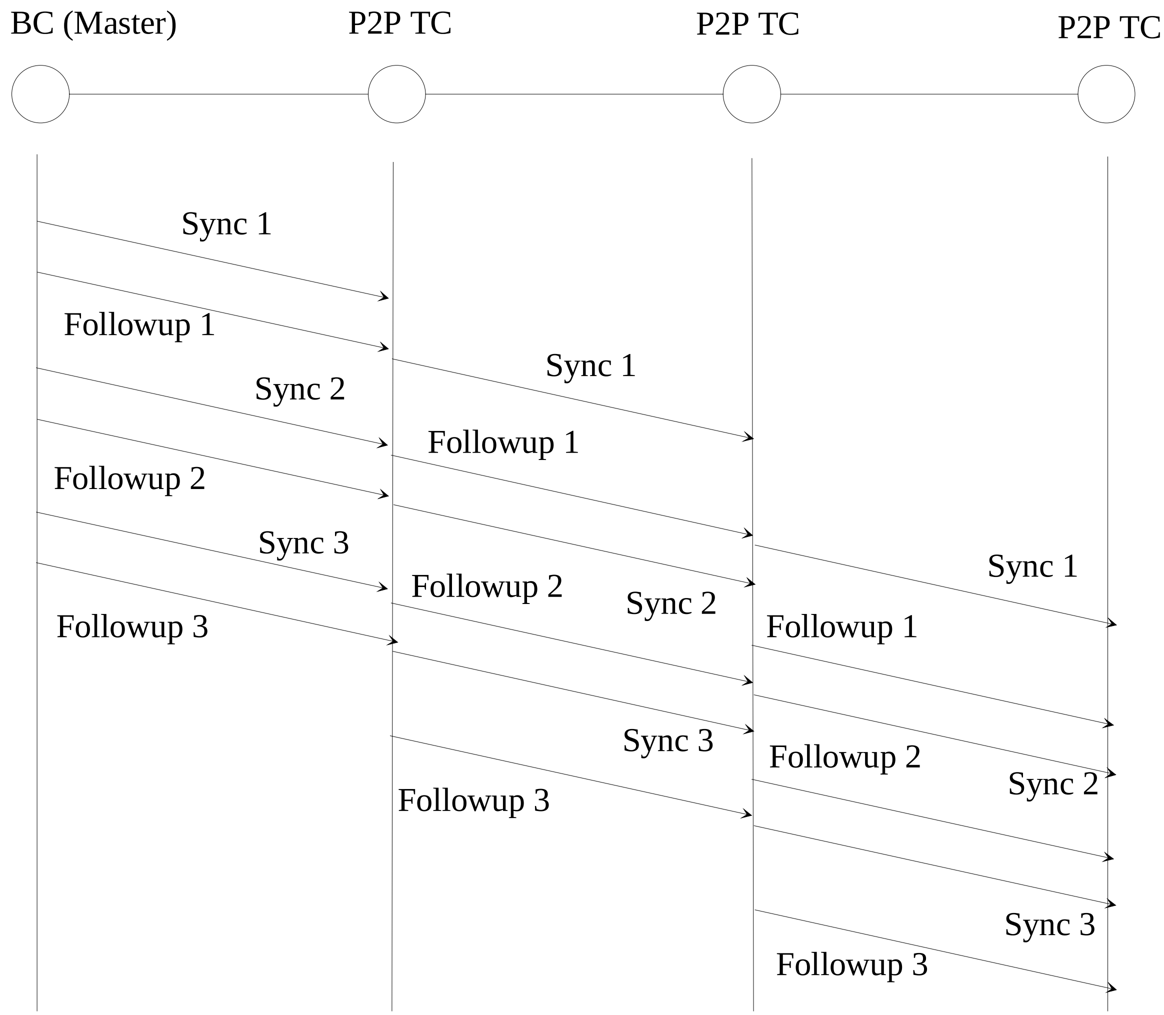


Note: The Followup messages corresponding to Sync2 and Sync3 are not shown to keep the diagram from being too cluttered.

Use of Sync and FollowUp with P2P TCs for AVB

- Multiple outstanding Sync messages could be avoided by
  - Increasing Sync interval to a value on the order of the FollowUp processing time multiplied by the maximum expected number of hops (i.e., to 70 ms in the above example)
  - Using a faster processor and decreasing the FollowUp processing time
  - Both approaches were considered undesirable when discussed in the 1588/AVB Design Meeting [3]
- Another approach was considered in the 1588/AVB Design Meeting [3]
  - When a Sync message arrives at a P2P TC, it is timestamped and held until the corresponding FollowUp message arrives
  - When the FollowUp message arrives, its correction field is added to the correction field of the Sync message, and the Sync message is sent and timestamped
  - Using the timestamps for when the Sync arrived and was sent, the residence time for the Sync is computed
    - This residence time will be much larger than the residence time in the conventional scheme; i.e., the residence time will be on the order of the FollowUp processing time
  - A new FollowUp message is generated, and the sum of the residence time for the Sync message at this P2P TC and the delay on the upstream link is placed in the correction field of this FollowUp message
  - This is illustrated on the next slide
    - The result is that when FollowUp arrives at a P2P TC or slave, there is only one outstanding Sync

Use of Sync and FollowUp with P2P TCs for AVB



Use of Sync and FollowUp with P2P TCs for AVB

- The description TCs in [1] and [2] is much more general than the description here; those references consider
  - P2P and E2E TCs
  - Followup and On-the-Fly TCs
  - BCs/OCs that cannot process ADelay messages (i.e., the upstream or downstream P2P TCs in this case must issue Delay_Req or Delay_Resp) and those that can
- In the current presentation, we have specialized to
  - P2P TCs
  - Followup TCs
  - BCs/OCs that can process ADelay messages
- In the description here for use in AVB networks, the semantics of the preciseOriginTimestamp, correction fields in the Sync and FollowUp messages, and the timestamps signifying when the Sync message is sent and arrives at the successive nodes is consistent with [1] and[2]

References

1. *Transparent Clock – Working Technical Description*, Version 12, IEEE 1588 TC Subcommittee, February 17, 2006.
2. *Revisions made to Transparent Clock – Working Technical Description, Version 12 at February 22 – 24, 2006 IEEE 1588 Face-to-Face Meeting* (see meeting minutes).
3. IEEE 1588/802.1 AVB Design Meeting, February 21, 2006 (see meeting minutes).
4. IEEE Std™ 1588-2002, *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE, November 8, 2002.
5. Geoffrey M. Garner, *Initial Description of Possible Use of Sync and FollowUp Messages with Peer-to-Peer Transparent Clocks in AVB*, Revision 1.0, Samsung contribution to IEEE 1588 and IEEE 802.1 AVB TG, March 1, 2006.