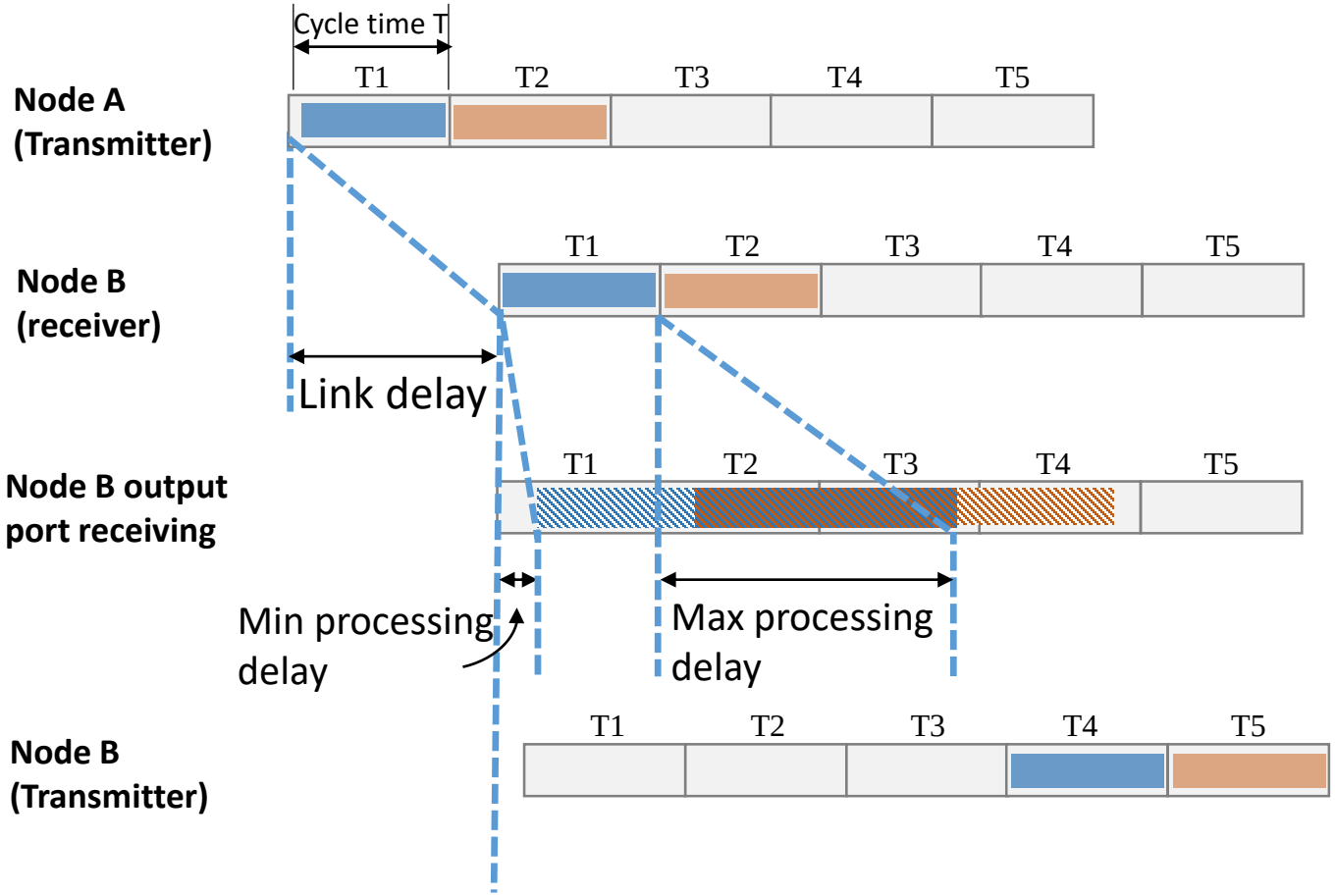
Labeling and software time stamp considerations in CQF enhancement

Yizhou Li (Huawei)
Guanhua Zhuang (Huawei)
Jiang Li (Huawei)
Li Dong (Shenyang Institute of Automation)
Wenbin Dai (Shanghai Jiao Tong University)

Recap – enhancement to CQF

- There are a number of contributions around the enhancement to CQF recently
  - Multiple CQF (802.1/dcn/21/1-21-0059-00.pdf/ 802.1/dcn/21/1-21-0059-01-ICne.pdf)
    - Use multiple instances of CQF on one port with different cycles
    - Specifies how exactly 3-buffer CQF works
    - Revised with more details like parameterization and managed objects in -01
  - Input sync for CQF (802.1/dcn/21/1-21-0056-00.pdf)
    - Use Time Marker Frame and CQF Phase Offset Msg to set the starting time of the next cycle of the downstream node to offset the long propagation delay
    - Address asymmetry link delay or SyncE-only use scenarios
  - Small cycle impact (docs2021/new-yizhou-small-cycle-impact-0914-v01.pdf)
    - When applying small cycle in CQF, internal processing variation introduces cycle ambiguity and >3 buffer requirement
    - Consider labeling to remove the cycle/buffer ambiguity
  - Pulsed queues (docs2021/new-finn-pulsed-queuing-0821-v03.pdf)
    - Use multiple bins in implementation in one priority queue
  - Non-FIFO queues (docs2021/new-specht-non-fifo-queues-0721-v01.pdf
    - Thoughts regarding syntonized CQF (issue discussions) and Paternoster (missing clean analytic proof)
  - Paternoster (docs2019/cr-seaman-paternoster-policing-scheduling-0519-v04.pdf)
    - Per-flow shaper on talker

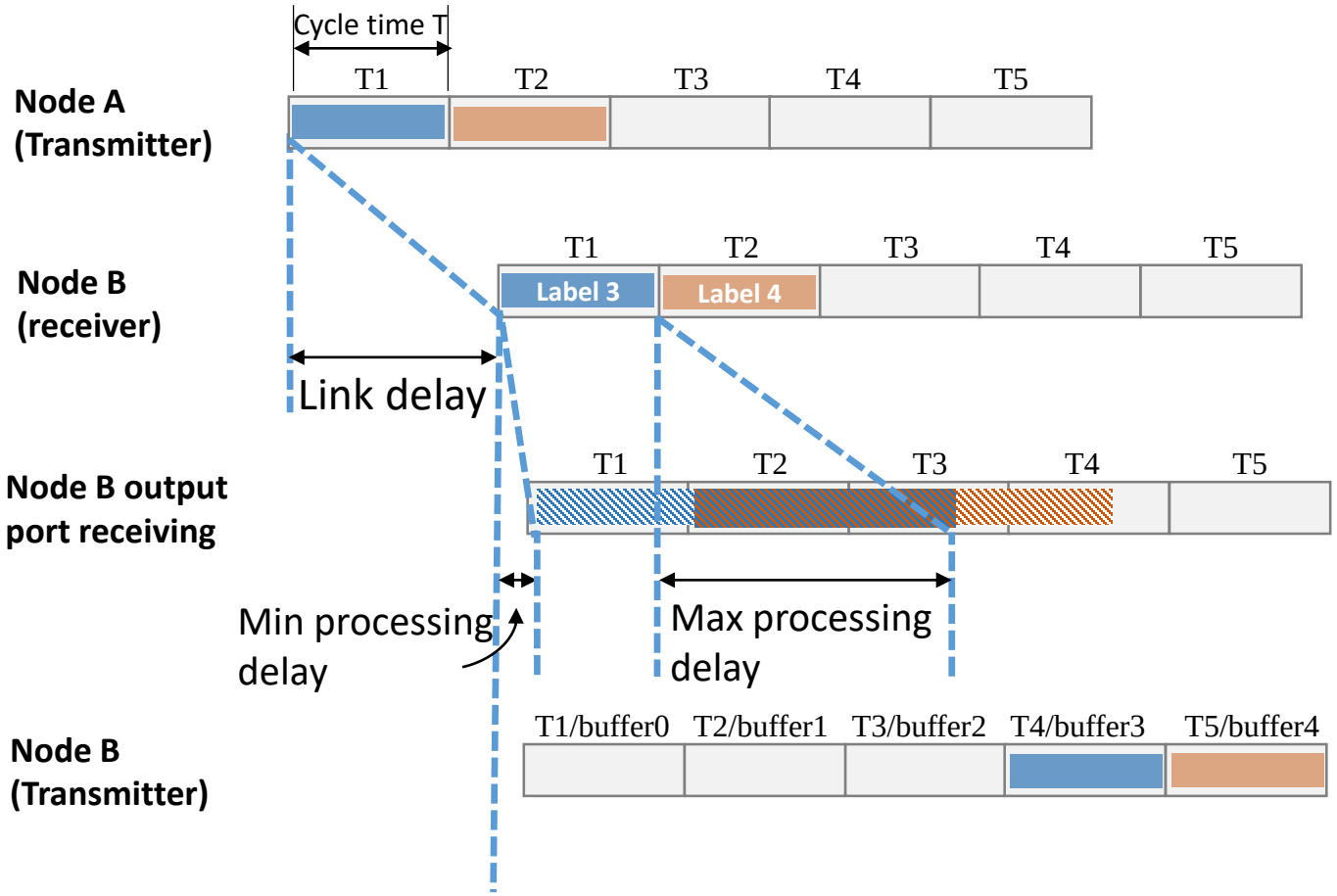
Internal Delay Variation and Internal Labeling



- Theoretically
  - link delay has no/negligible variation
  - The latency from Node A transmitter to Node B transmitter is composed of
    - Static delay: link delay
    - Dynamic delay: node processing delay + wait time
- Cycle/buffer ambiguity mainly comes from the node processing delay variation between input port and regulator at Node B (*).
- To remove the cycle/buffer ambiguity inside a node, internal labeling can be added at the input port to distinguish the output port buffer a packet should go.

(*) new-yizhou-small-cycle-impact-0914-v01

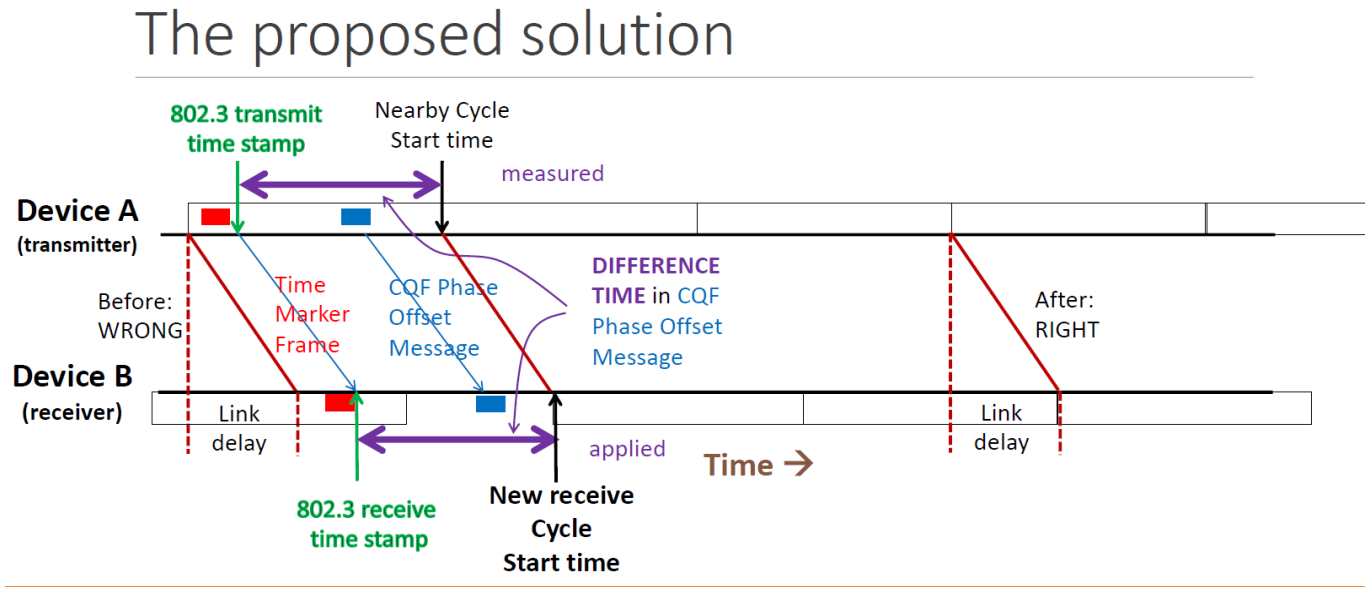
Use internal labeling to remove the ambiguity caused by internal delay variation



- At Node B's receiver, demarcation is time based because no mixture of packets from different cycles at A can be received within the same cycle at B's input port.
- Label the packets based on such demarcation, i.e. based on the cycle time
- Internal label maps to one of the cyclic buffers at Node B's output port. (Mapping relationship is pre-computed given the processing delay variation)

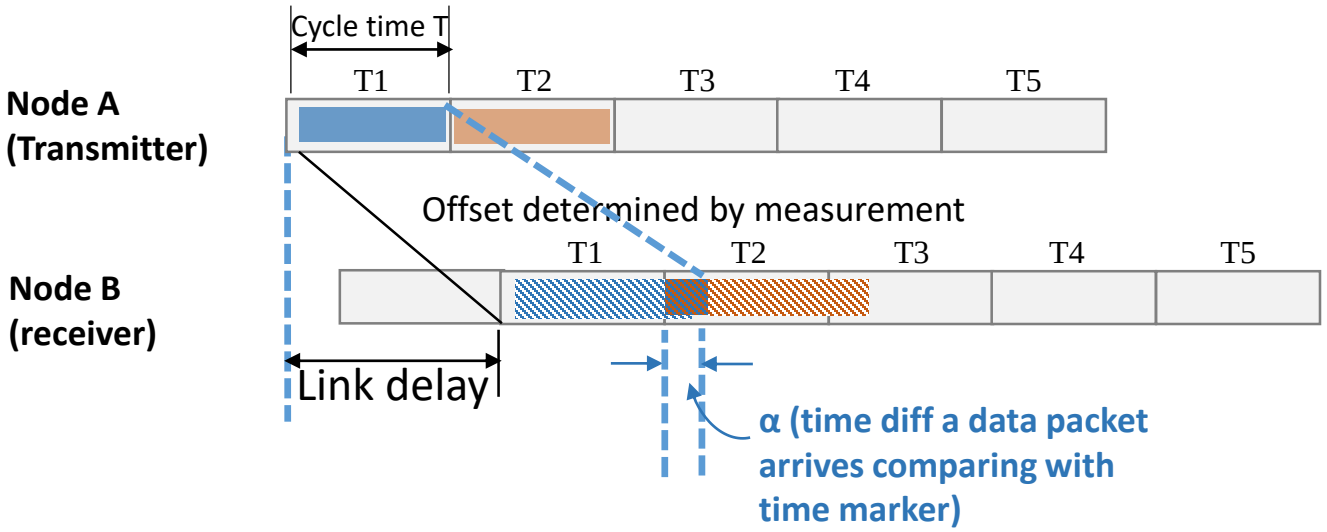
- **In context of 802.1Q:**
  - Keep using IPV as internal labeling. (32-bit signed integer is sufficient for total number of buffers)
  - Clarification to be added to more clearly explain why >3 buffer is required and IPV is used for time based packet demarcation to avoid cycle ambiguity inside a node.

Revisit the link delay measurement impact



- A hardware timestamping based approach was proposed in 1-21-0056-00 (Input synchronization in CQF) slide 35.
- The real data packet propagation delay is always larger than the measured link delay based on hardware time stamps.
- The output variation is the time taken from gate opening indicated by GCL (gate control list) to the transmission of the first bit of the packet on the physical link.
- **Some preliminary testing**: output variation is 0~8 usec

When output variation α is introduced



- Potential problems:
  - Time based demarcation of packets at Node B's receiver has ambiguity.
  - Some packets of blue cycle may be labelled with (L+1) instead of L

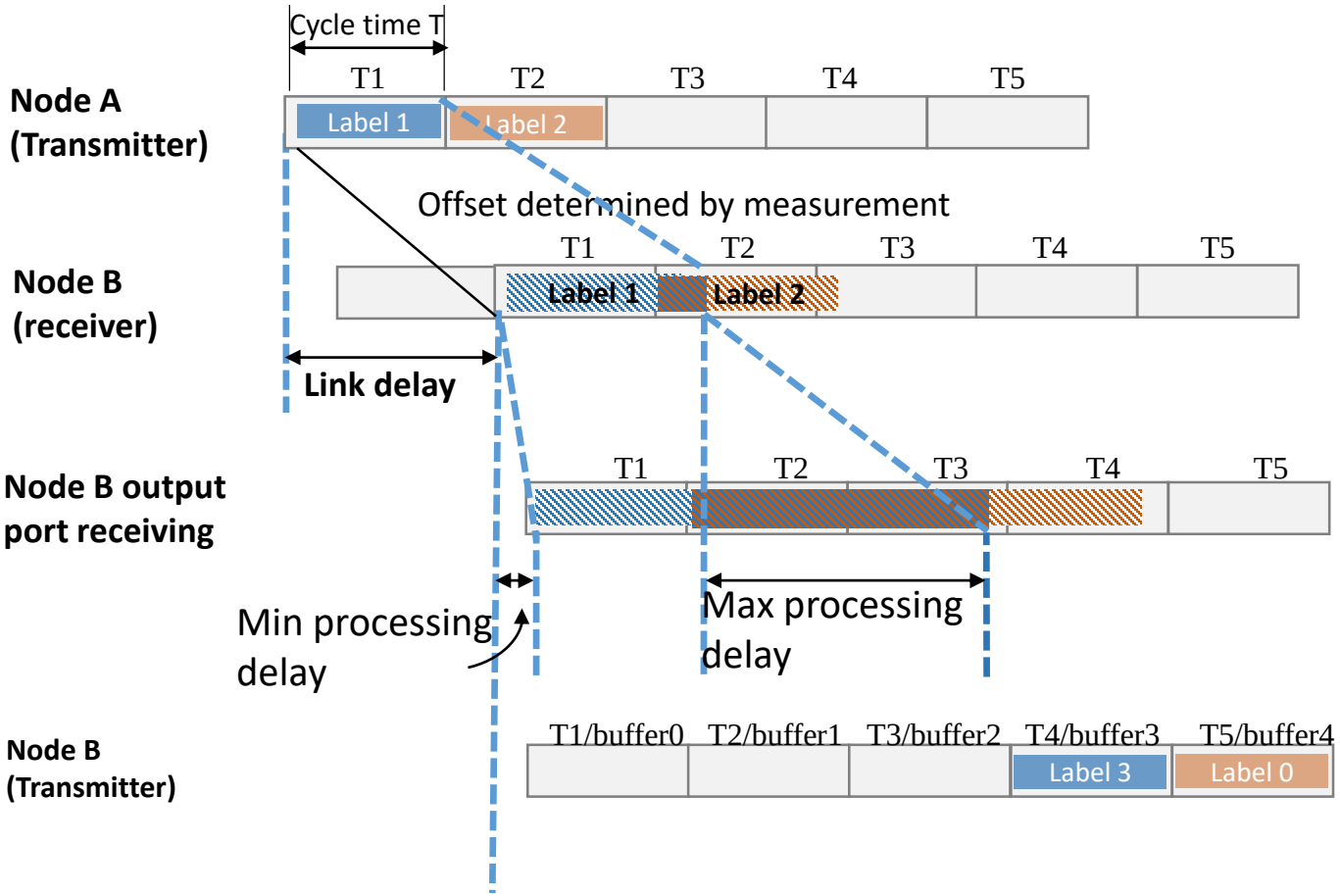
- Current way to solve it: increase dead time by α at the end of the cycle at Node A's transmitter.

A second thought

- Dead time can always work as a cure-all for any kinds of unknown/unexpected variations.

- The potential problems however:
  - When the cycle time is small (31.25 usec or less), ~8 usec dead time caused by output variation is not acceptable. It eats the cycle time up.
  - Dead time is better to be used as the last resort rather than a cure-all.
    - It is an accumulated value from multiple variation factors. Accurately measuring each contributing factor is not possible.
    - To play safe, each contributing factor to the dead time normally has to be over-estimated.
    - Value guessing like configuration is a burden for network admin.
    - The empirical values have to be provided usually. What are the values?

Another way: use external labeling to remove the ambiguity introduced by external variation



- At Node A's transmitter
  - Packets are labelled so that demarcation of packets are fixed
- At Node B's receiver, label_in maps to one of the cyclic buffers at Node B's transmitter with a label_out. (Mapping relationship is pre-computed)
  - In implementation, IPV can be used internally to indicate the cycle demarcation/buffer and then further binds to a label_out

Label_in	Label_out	IPV
1	3	**6**
2	0	**7**
3	1	**4**
0	2	**5**

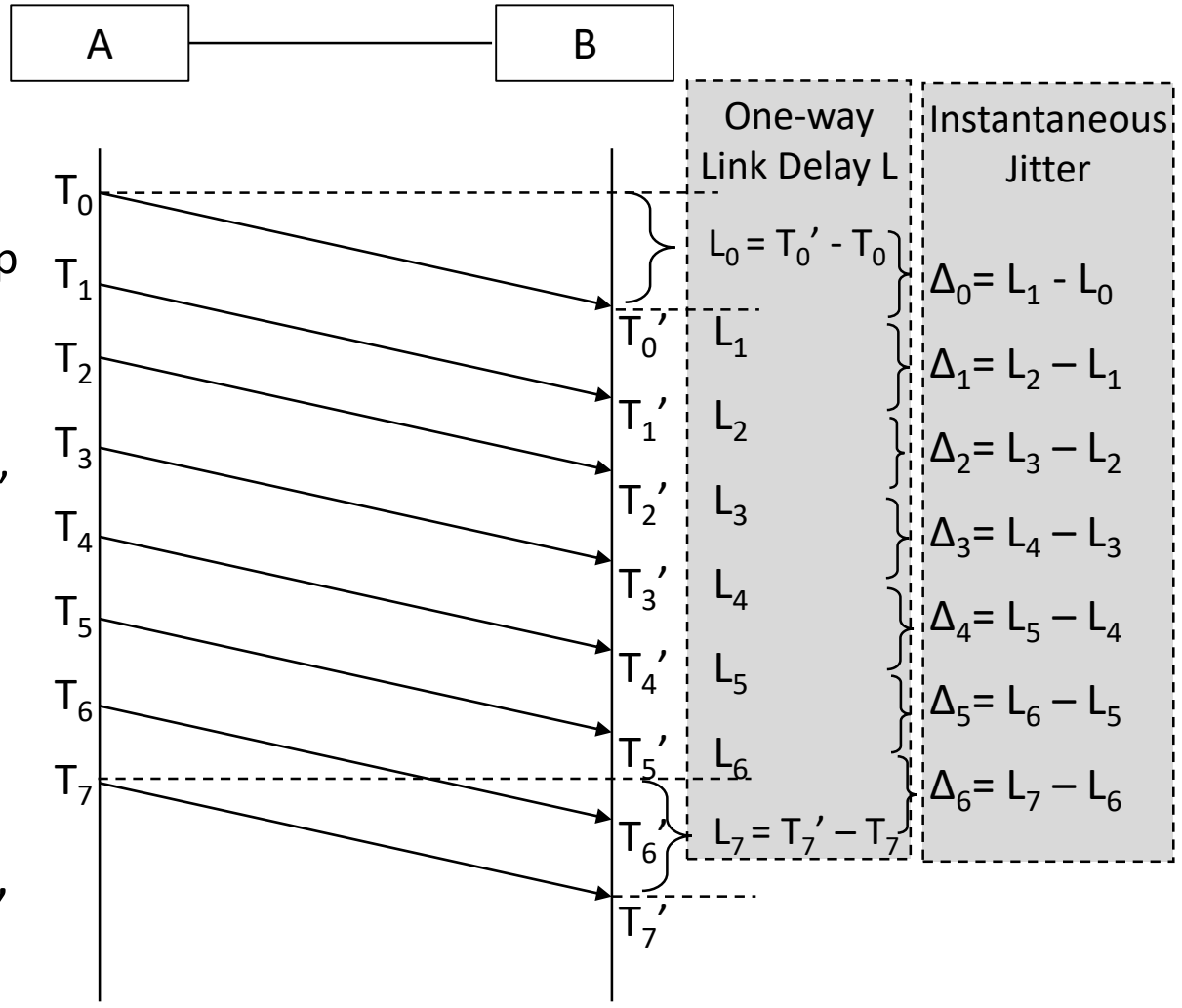
- The only change to measurement message is that TMF (time marker frame) need carry label.

A question to discuss

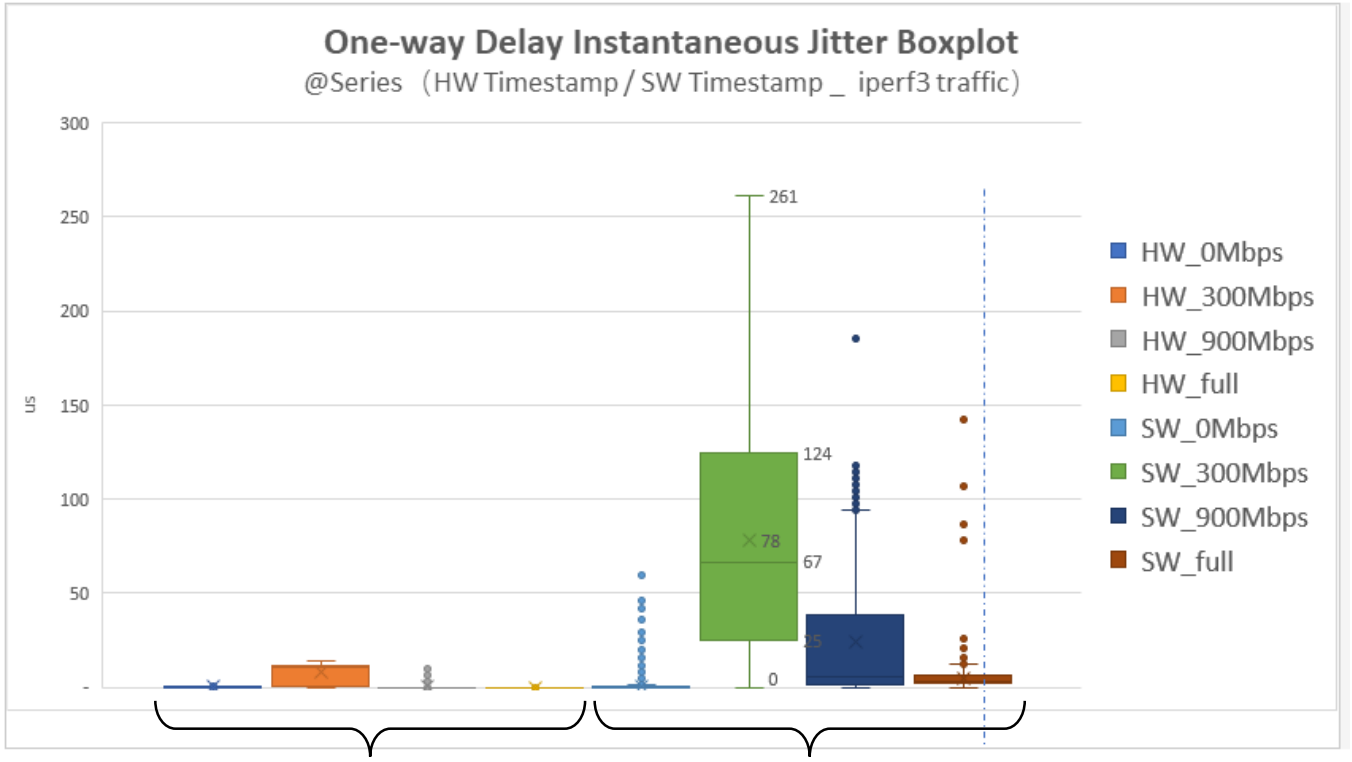
- Can we directly migrate the measurement mechanism to be software timestamping based?
- Why it may be required?
  - Time Synchronization Service Interface (TSSI) in IEEE Std 802.3 Clause 90 is optional and most likely implemented when time synchronization is there.
  - Some syntonization (instead of synchronization) mechanisms like SyncE does not use TSSI. The support to get the accurate PHY transmitting time is not available.
  - Extra cost for Clock pinch board.
  - Some network nodes and end hosts implement time sync protocol as software.

Evaluation on the software timestamping based link delay instantaneous jitter

- Experiment setup
  - 1Gbps link between two nodes A & B.
  - A & B are not frequency synchronized (because we did not find the right equipment)
  - A timestamps T_n in software and B uses local timestamp T_n' to get one-way link delay with time drift L.
  - L is subject to accumulative time drift effect. So we use instantaneous jitter Δ, i.e. the difference of two consecutive values of L. If link delay measured is stable, Δ should be almost 0.
  - Use the proprietary UDP packet. Size is 212B on wire.
  - Send 5K packets with timestamp every second.
  - Run 50K times for each testing case
- Compare using hardware and software based timestamp to measure the variation of Δ under different background traffic (0, 300Mbps, 900Mbps, full).



Test results



Hardware timestamping

Software timestamping

- The hardware (HW) timestamp is stable irrespective of background traffic
- The software (SW) timestamp has more obvious value outliners
- The variation of Δ is the most significant when background traffic is 300Mbps because the change of one-way delay can be dramatic due to burst.
- When the background traffic goes higher, it becomes the stable high bandwidth consumption rather than burst. The variation of Δ is lower on the contrary. (Δ is instantaneous jitter)
- What would be required if software timestamp is used:
  - Remove outliners
  - Smooth the adjustment based on measurement
  - Dual message style of Sync & Sync follow up provides no extra accuracy. Cycle marker frame at the start of a cycle may be revisited to see the applicability.

Suggestions

- Clarification to be added to more clearly explain why >3 buffer is required.
- Clarify IPV used as the internal label for time based packet demarcation to avoid cycle ambiguity inside a node.
- If concept of “bin” is going to be introduced, clarify the IPV indication to bin/queue
- Use the external labeling as an optional mechanism to alleviate the dead time consumption and improve utilization
- Consider to allow the use of software based time stamps for measurement. More details TBD.