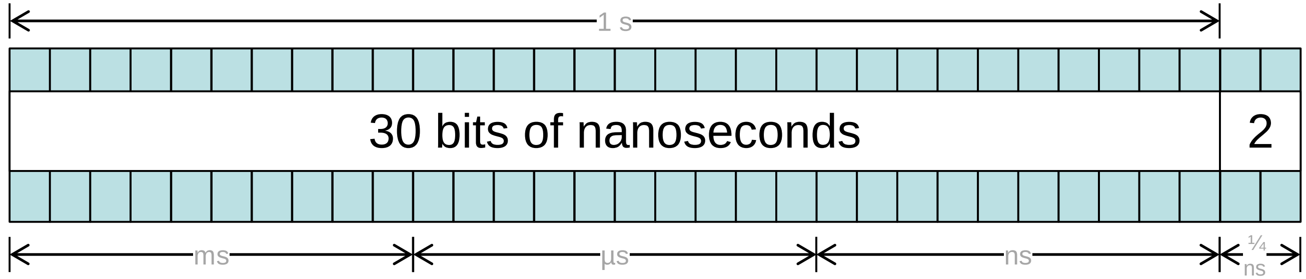
Timestamp Precision

Kevin Bross

15 April 2016

Timestamp Packets

- The *orderInfo* field can be used as a 32-bit timestamp, down to ¼ ns granularity:



- Two main uses of timestamp:
  - Indicating start or end time of flow
  - Indicating presentation time of packets for flows with non-constant data rates

s = second
ms = millisecond
µs = microsecond
ns = nanosecond

Presentation Time

- To reduce bandwidth during idle periods, some protocols will have variable rates
  - Fronthaul may be variable, even if rate to radio unit itself is a constant rate
- Presentation times allows RoE to handle variable data rates
  - Data may experience jitter in network
  - Egress buffer compensates for network jitter
  - Presentation time is when the data is to exit the RoE node
    - Jitter cleaners ensure data comes out cleanly, and on the right bit period

Jitter vs. Synchronization

- Synchronization requirements for LTE are *only* down to ~±65 ns accuracy
  - Each RoE node may be off from TAI by up to 65 ns (or more in some circumstances)
  - Starting and ending a stream may be off by this amount
- …but jitter from packet to packet must be much tighter
  - RoE nodes should be able to output data at precise relative times if timestamp is used for a given packet
  - Relative bit time within a flow is important

Timestamp Precision

- Is the current ¼ ns granularity tight enough for today’s systems, and does it have headroom for the future?
  - Each bit in 9.8 Gbps CPRI is ~1/10 ns
  - Each bit in 24 Gbps CPRI is ~1/24 ns
  - Rates of 100 Gbps or more are likely in the reasonable future

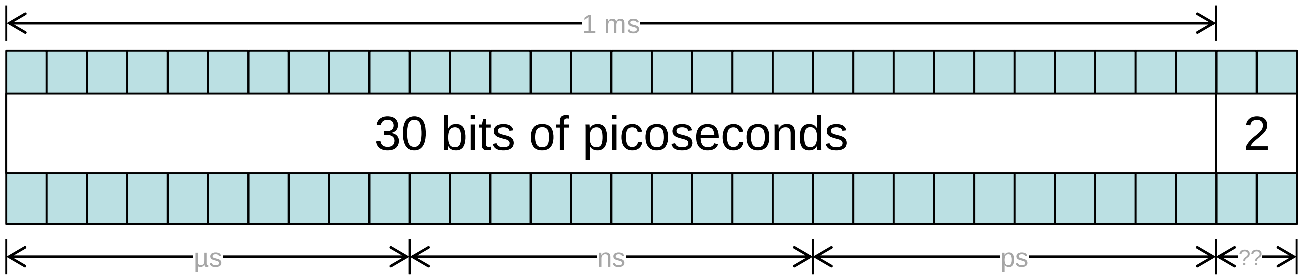
- How do you specify a presentation time with bit times that may be tiny fractions of nanoseconds if the smallest unit is in ¼ nanoseconds?

Hypothetical Example

- Assume 100 Gbps raw data rate, with extended idle periods suppressed
- Raw data:
  - …, 0x3F, 0x4E, <807 bytes of 0's>, 0x39, 0x41
  - One packet ends with 0x4E
  - Next packet starts with 0x99
- How does RoE say when that packet is supposed to hit?
  - One bit position early, first byte = 0x72
  - One bit position late, first byte = 0x1C
- Relative timing of bits is important

Proposed Redefinition

- Define timestamps in terms of picoseconds, rather than nanoseconds:

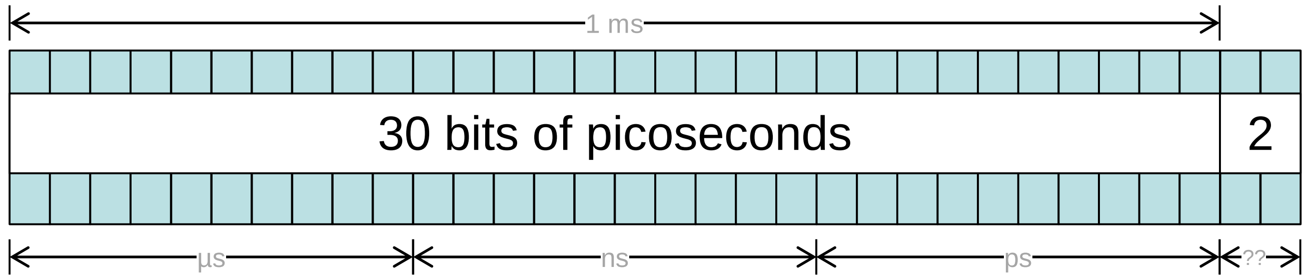


- Justification:
  - Prior discussions indicated transit time will be less than 1 ms now and in future
  - Provides headroom for future speed increases
  - No change in field length or usage

ms = millisecond
µs = microsecond
ns = nanosecond
ps = picosecond

Lower 2 Bits

- 30 bits will handle down to picosecond level—what about the last 2 bits?

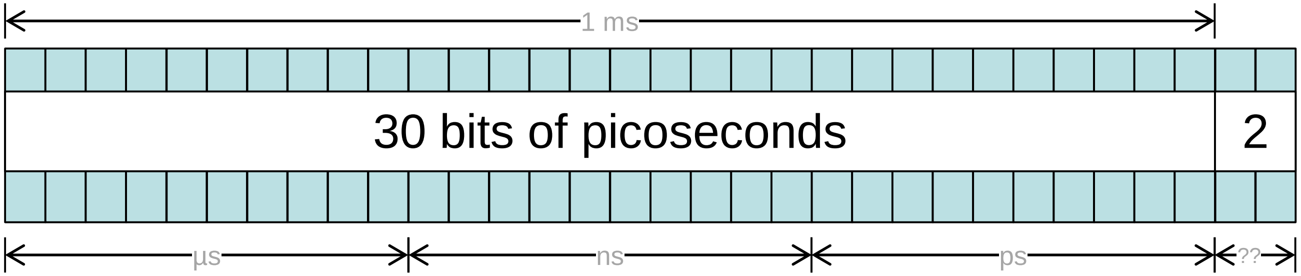


- Two options (the first is recommended):
  - Implement mini sequence number (0, 1, 2, 3, 0, 1, 2, 3, …) to detect missed packets
    - Get some benefits of sequence number w/timestamp
  - … or continue down to ¼ ps time

ms = millisecond
µs = microsecond
ns = nanosecond
ps = picosecond

Summary

- This presentation proposes re-defining timestamp to handle picosecond timing:



  - This handles presentation times up to 1 ms in the future, while offering precision to 1 ps
  - Lower 2 bits could implement 2-bit sequence number to detect the occasional missing packet
    - Or could be used to go down to ¼ ps timing
  - Timestamp purpose/usage unchanged

ms = millisecond
µs = microsecond
ns = nanosecond
ps = picosecond