Interleaved Synchronization Protocols for LANs and Space Data Links

Sir John Tenniel; *Alice's Adventures in Wonderland,*Lewis Carroll

David L. Mills
University of Delaware
http://www.eecis.udel.edu/~mills
mailto:mills@udel.edu

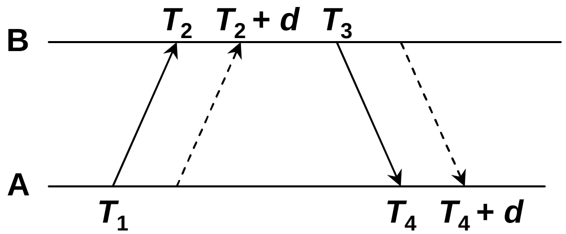
Introduction

- This presentation suggests modifications for three time synchronization protocols used on various LANs and space data links.
  - Network Time Protocol (NTP) used for Internet synchronization with potential accuracies in the low microseconds range.
  - IEEE 1588 Precision Time Protocol (PTP) used for hardware synchronization with potential accuracies in the low nanoseconds range.
  - Consultive Committee on Space Data Systems (CCSDS) Proximity-1 Time Services Protocol used for Mars space data links with potential accuracies in the tens of nanoseconds range.
- The modifications provide improved performance and reduced complexity using an interleaved design where the transmit timestamp is transmitted in the following packet.
- The presentation covers each of these protocols in turn.
- This briefing is based on the white paper Analysis and Simulation of the NTP On-Wire Protcol.

NTP interleaved symmetric and broadcast protocol

- In principle, NTP can deliver submicrosecond accuracy if timestamps can be captured precisely.
- Current performance of a primary server with GPS reference clock and PPS signal is typically 2-5 μs.
- Current performance of a secondary server relative to a primary server is 20-50 μs on a fast LAN with 16-s poll interval.
- We would like to improve the performance for a secondary server to the level of the PPS signal.
  - Capture the timestamps closer to the transmission media.
  - These timestamps might not be available to include in the packet, as in current NTP.
- Modify the NTP on-wire protocol to accommodate late timestamps while preserving backwards compatibility and without changing the NTP packet format.

Software timestamps (NTP)



- Software timestamps as used by NTP.
  - Transmit timestamps are captured shortly before the beginning of the packet; receive timestamps are captured shortly after the end of the packet.
  - Assume *d* is the packet tranmission time.
  - offset $\theta = ½\{[(T_2 + d) – T_1)] + [T_3 – (T_4 + d)]\}$ so *d* cancels out.
  - delay $\delta = [(T_4 + d) – T_1] – [T_3 – (T_2 + d)]$ so *d* cancels out.
- Conclusion: If the delays are reciprocal and the packet lengths the same, software timestamps are equivalent to hardware timestamps.
- Any other combination has errors depending on *d*.
- Further information is at Timestamping Principles.

NTP protocol header and timestamp formats

NTP Protocol Header Format (32 bits)

LI	VN	Mode	Strat	Poll	Prec
Root Delay					
Root Dispersion					
Reference Identifier					
Reference Timestamp (64)					
Origin Timestamp (64)					
Receive Timestamp (64)					
Transmit Timestamp (64)					
Extension Field 1 (optional)					
Extension Field 2… (optional)					
Key/Algorithm Identifier					
Message Hash (64 or 128)					

Cryptosum

Authenticator (Optional)

LI	leap warning indicator
VN	version number (4)
Strat	stratum (0-15)
Poll	poll interval (log2)
Prec	precision (log2)

NTP Timestamp Format (64 bits)

Seconds (32)	Fraction (32)

Value is in seconds and fraction since 0^h 1 January 1900

NTPv4 Extension Field

Field Length	Field Type
Extension Field (padded to 32-bit boundary)	

Last field padded to 64-bit boundary

NTP v3 and v4
NTP v4 only
authentication only

Authenticator uses DES-CBC or MD5 cryptosum of NTP header plus extension fields (NTPv4)

NTP interleaved on-wire protocol

- The primary purpose of the interleaved on-wire protocol is to improve accuracy using driver timestamps (drivetstamps) or hardware timestamps (hardstamps).
- Another purpose is when the message digest is computed by a separate secure process, as in Microsoft Active Directory.
- It is an extension of the current NTP on-wire protocol and is backwards compatible with it, including resistance to lost, duplicate or bogus packets.
- It operates in basic, interleaved symmetric and interleaved broadcast modes and automatically adapts to normal or interleaved operation.
- As in the current design, the protocol accumulates four timestamps in each round.
  - Symmetric peers use these timesatmps to determine offset and delay of each relative to the other.
  - Broadcast clients determine delay in the first round and then revert to listen-only.

Basic and interleaved protocol state variables

- State variables
  - *xmt* transmit timestamp
  - *rec* receive timestamp
  - *dst* destination timestamp
  - *aorg* alternate origin timestamp
  - *borg* alternate origin timestamp
  - *x* toggle switch (+1, 0, -1)
  - *f* synch bit (0 or 1)
  - *b* broadcast bit (0 or 1)
- Packet header variables
  - t_{org} origin timestamp
  - t_{rec} receive timestamp
  - t_{xmt} transmit timestamp

Transmit and receive protocol state machines

- The following slides show the flow charts of the state machihes that implement the basic and interleaved variant of the various modes.
  - Slide 1: transmit process used in all modes
  - Slide 2: receive process for basic and interleaved broadcast modes
  - Slide 3. receive process for basic symmetric mode
  - Slide 4. receive process for interleaved symmetric mode
  - Slide 5. receive process for timestamp checking

Transmit process

```
if (mode != BCST) { /* broadcast */
    t_org = rec
    t_rec = dst
    if (x == 0) {   /* basic */
        aorg = clock
        t_xmt = aorg
    } else {         /* interleaved */
        if (x > 0) {
            aorg = clock
            t_xmt = borg
        } else {
            borg = clock
            t_xmt = aorg
        }
        x = -x
     }
} else {
    t_org = aorg
    aorg = clock
    t_rec = 0
    t_xmt = aorg
}
```

Receive process – broadcast modes

```
err = OK
if (t_xmt != 0 && t_xmt == xmt) {
    err = DUPE
} else if (mode == BCST) {   /* broadcast */
    xmt = t_xmt
    if (t_org == 0)   {          /* basic */
        dst = clock
        T_3 = t_xmt
        T_4 = dst
    } else {                     /* interleaved */
        T_3 = t_org
        T_4 = borg;
        if (T_4 == 0)
            err = SYNC       /* unsynchronized */
        else if (t_org - aorg > MAX)
            err = DELY       /* delay error */
        aorg = t_xmt
        dst = clock
        borg = dst
}
(continued)
```

Receive process – basic symmetric mode mode

```
} else if (x == 0) {   /* basic symmetric */
    xmt = t_xmt
    rec = t_xmt
    dst = clock
    T_1 = t_org
    T_2 = t_rec
    T_3 = t_xmt
    T_4 = dst
    if (T_1 == 0 && T_2 == 0 && T_3 != 0)
        err = SYNC     /* unsynchronized */
    else if (T_1 == 0 || T_2 == 0 || T_3 == 0)
        err = ERRR     /* protocol error */
    else if (T_1 != aorg)
        err = BOGUS    /* bogus */
(continued)
```

Receive process – interleaved symmetric mode

```
} else {            /* interleaved symmetric */
    xmt = t_xmt
    if (x > 0)
        T_1 = aorg
    else
        T_1 = borg
    T_2 = rec
    T_3 = t_xmt
    T_4 = dst
    rec = t_rec
    dst = clock
    if ((t_org == 0 && t_rec == 0 && t_xmt == 0)
      || (t_org == 0 && t_rec != && t_xmt != 0)) {
        f = 1; err = SYNC     /* unsynchronized */
    } else if (f == 0) {
        reset(); err = HOLD   /* hold off */
    } else if (t_rec == 0 || t_xmt == 0)
        reset(); err = ERRR   /* protocol error */
    } else if (T_2 == 0) {
        err = SYNC            /* unsynchronized */
    } else if (t_org != T_4)
        reset(); err = BOGUS /* bogus */
    }
(continued)
```

Receive process – timestamp checking

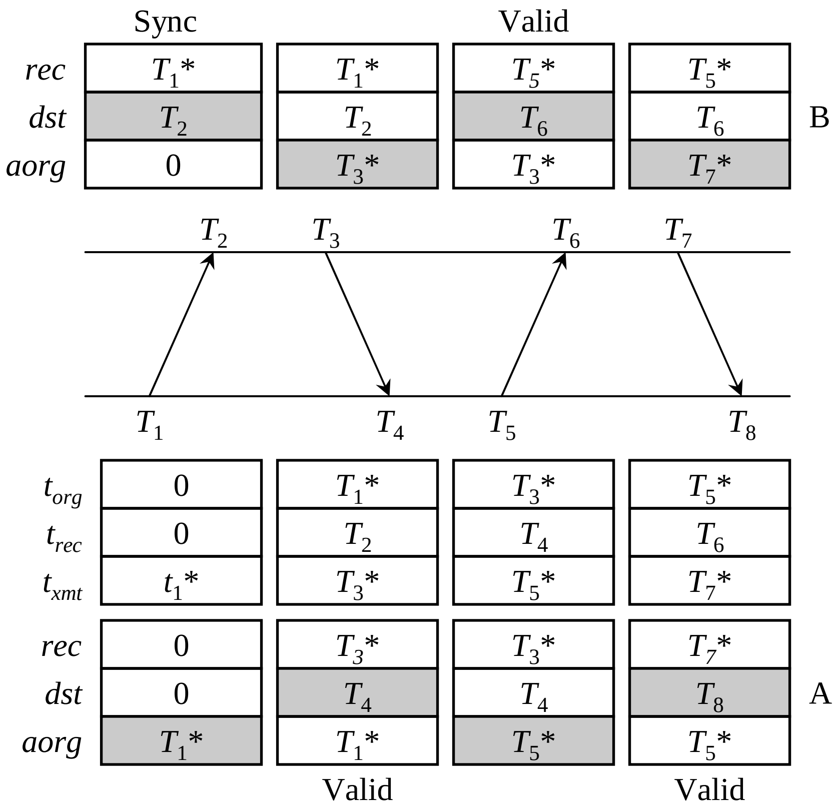
```
    if (err == OK) {
        d = (T4 - T1) - (T3 - T2)

        if (T3 > T2 || T1 > T4)
            err = INVL     /* invalid timestamp */
        else if (d < 0 || d > 1)
            err = DELY     /* delay error */
        else if {abs(T2 - T1) > MAX || abs(T3
          - T4) > MAX)
            err = OFST;    /* offset error */
    }
}
```

NTP basic on-wire protocol

- The following figure shows two rounds of the protocol.
  - The transmit timestamps carry odd subscripts while the receive timestamps carry even subscripts.
  - Packets are transmitted along the direction of the arrows.
  - Timestamps are captured from the clock in the blue boxes. They are copied from there to other state variables and packet headers.
- At T_4 the first A round is complete and the timestamps T_1, T_2, T_3 and T_4 are available to compute offset and delay of B relative to A as described in the architecture briefing.
- At T_6 the first B round is complete and the timestamps T_3, T_4, T_5 and T_6 are available to compute the offset and delay of A relative to B.
- Operation continues in subsequent rounds.

Basic symmetric mode



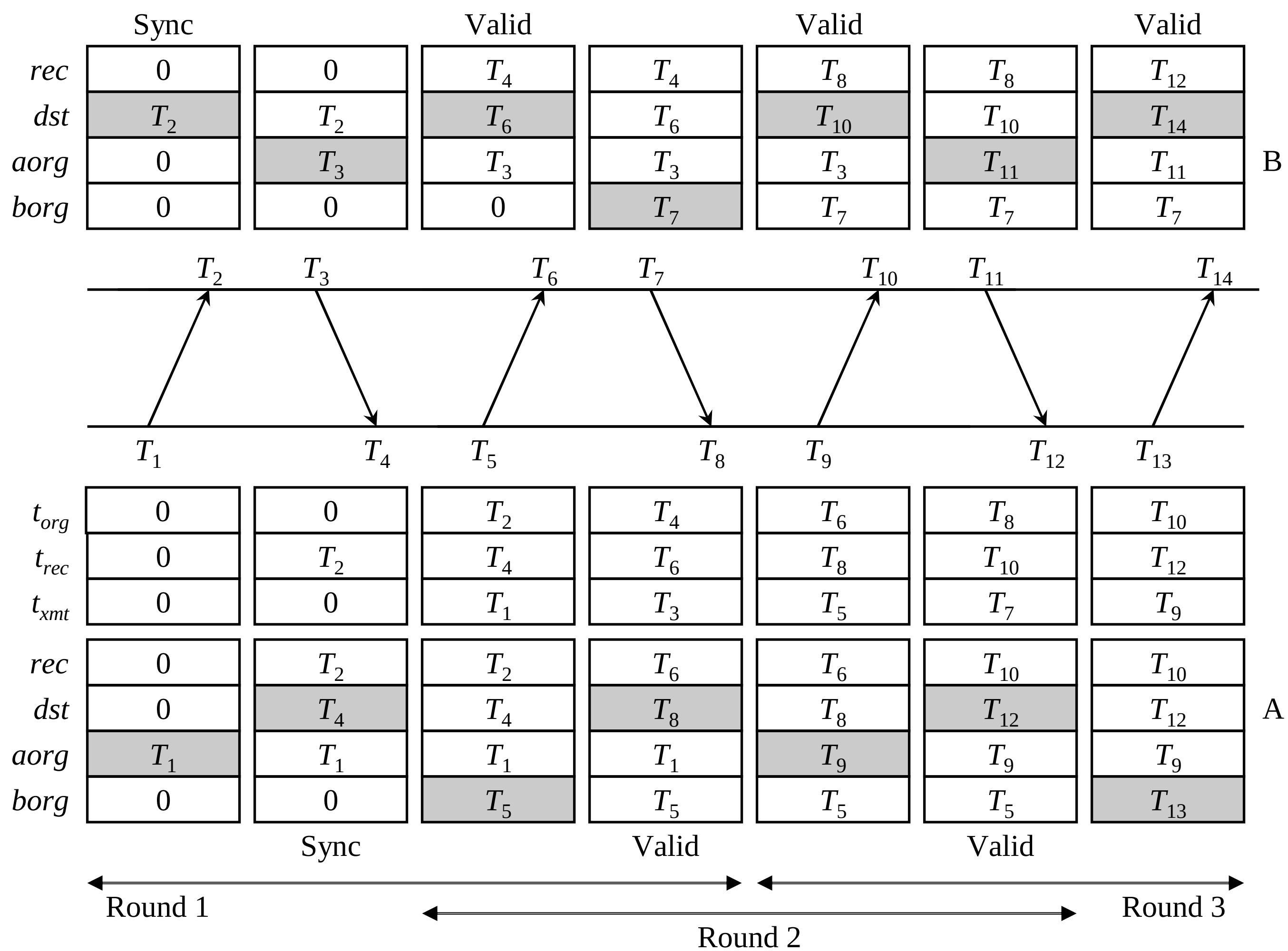
Timestamping principles

- Accuracy is diminished in the basic protocol because the elapsed time between the transmit softstamp and the drivestamp determined by the interrupt routinecan be significant.
- A more accurate transmit drivestamp could be captured by the NIC driver or better yet a hardstamp captured by the hardware PHY.
- However, doing that means the transmit timestamp is not available to include in the packet.
- The solution is to include the transmit timestamp in the following packet.
- The trick is to do this using the same NTP packet header format and to automatically detect whether basic or interleaved mode is in use to support past protocol version.

Interleaved on-wire protocol

- The interleaved protocol uses five state variables, *rec*, *dst*, *aorg*, *borg* and *xmt* for each peer. The *xmt* variable is used only to detect duplicate packets and is not shown in the figures.
- The protocol requires two basic rounds to produce the timestamps that determine offset and delay; however, the rounds are interleaved so that one set of timestamps is produced for each basic round.
- A new transmit softstamp and hardstamp is produced for each transmitted packet, but the softstamp is overwritten by the hardstamp before being sent.
- Each transmitted packet contains the previous transmit hardstamp.
- Once synchronized, the first set of timestamps t_1, t_2, t_3 and t_4, are available at t_6 and the next set at t_3, t_4, t_5 and t_6 at t_8 and so forth.

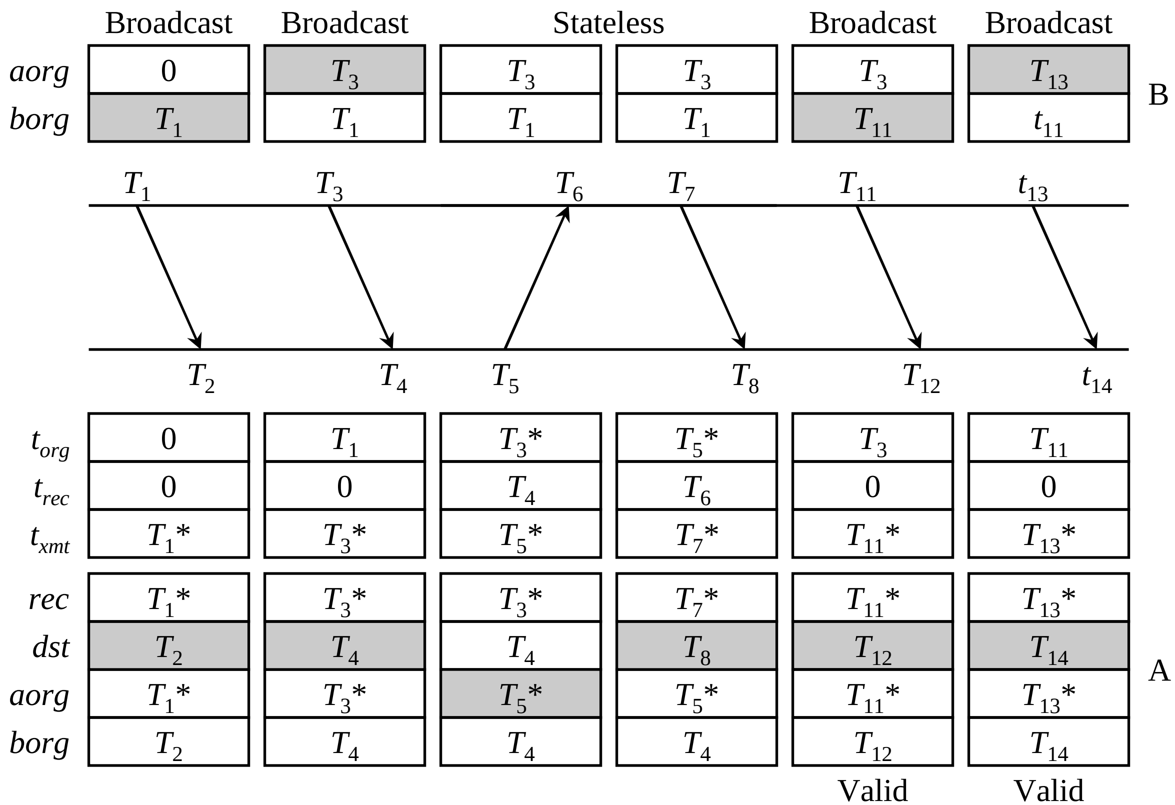
Interleaved symmetric mode



Interleaved on-wire broadcast protocol

- Interleaved broadcast is similar to IEEE 1588 two-step multicast, but does not require a follow-up message.
- The basic principle is that the transmit drivestamp for one broadcast packet is sent in the next broadcast packet. The roundtrip delay is determined in client-server mode, but with the opposite offset sign.
- The variant shown on the next slide is backwards compatible with current NTP. The timestamps with asterisks are captured before transmitting the packet, but are not used.
- The actual offset and delay is calculated as each broadcast packet arrives. The delay is saved for intervals when the stateless exchange is not used.
- In this figure softstamps and timestamps derived from them are shown with asterisk (*).

Interleaved broadcast mode

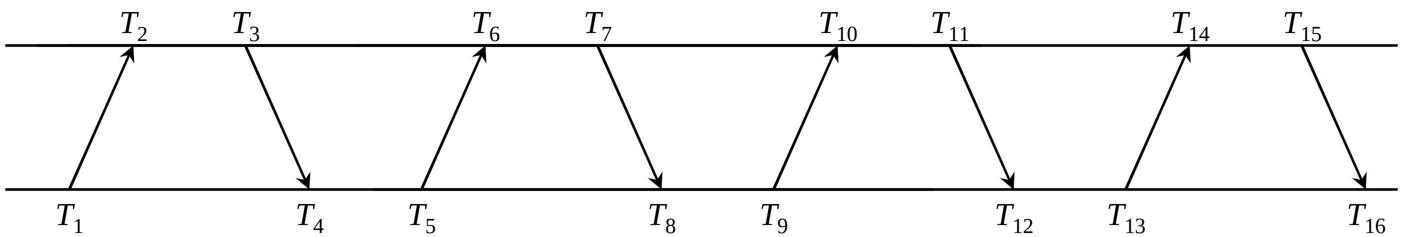


Automatic protocol detection

- The next slide shows how the protocol can detect whether the interleaved protocol is supported and, if not, how it can revert to basic mode.
- Peer B starts in interleaved mode; peer A client starts in basic mode and cannot switch to interleaved mode.
- Both client and server bungle on until the B detects an error at T_{10} and switches to basic mode. After synchronizing, operation continues in basic mode for both B and A.
- A simulator program to generate and test the protocol is available See Appendix B of Analysis and Simulation of the NTP On-Wire Protocol.

Automatic protocol detection example

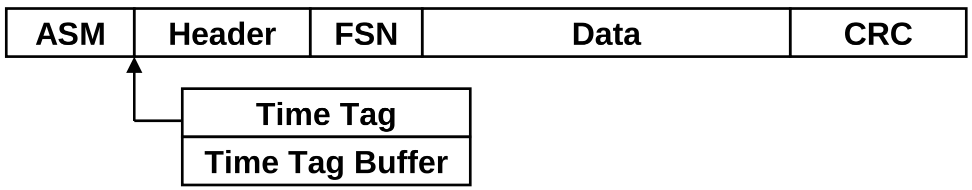
	Sync	Xleave	Sync	Xleave	Bogus	Basic	Valid	Basic	
rec	0	0	T_4	T_4	T_8	T_8	T_{13}*	T_{13}*	
dst	T_2	T_2	T_6	T_6	T_{10}	T_{10}	T_{14}	T_{14}	
aorg	0	T_3	T_3	T_3	T_3	t_{11}*	T_{11}*	T_{15}*	B
borg	0	0	0	T_7	T_7	T_7	T_7	T_7	



t_{org}	0	0	0	T_4	T_3	T_8	T_{11}*	T_{13}*	
t_{rec}	0	T_2	T_4	T_6	T_8	T_{10}	T_{12}	T_{14}	
t_{xmt}	T_1*	0	T_5*	T_3	T_9*	T_{11}*	T_{13}*	T_{15}*	
rec	*0*	0	0	T_3	T_3	T_{11}*	T_{11}*	T_{15}*	
dst	0	T_4	T_4	T_8	T_8	T_{12}	T_{12}	T_{16}	A
aorg	T_1*	T_1*	T_5*	T_5*	T_9*	T_9*	T_{13}*	T_{13}*	
borg	0	0	0	0	0	0	0	0	
	Basic	Sync	Basic	Bogus	Basic	Bogus	Basic	Valid	

Proximity-1 original time service protocol

Proximity-1 Frame



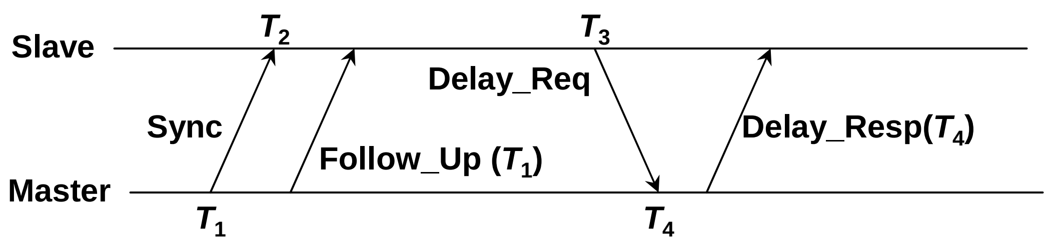
ASM Attached Sequence Marker
FSN Frame Sequence Number
CRC Cyclic Redundancy Check

- Proximity-1 protocol is used for Mars orbiter and rover data links.
- On command, the orbiter and rover time-tag the ASM for a number of transmitted and received frames and collect them and the associated FSNs in a buffer..
- The contents of the buffers are sent, perhaps via relay, to Earth.
- On Earth the transmit time-tags are matched with the respective receive time-tags and the spacecraft clock data to determine the offset of one spacecraft relative to the other.
- If necessary, the respective times are uploaded to the orbiter for relay to the rover.

Proximity-1 Interleaved Time Service (PITS)

- We propose a new Timestamp SPDU at each end of the space data link. It carries three 64-bit timestamps as in the NTP packet header.
- This requires a minor modification of the Proximity-1 radio to capture time-tags for the transmit and receive SPDUs. These will later be converted to logical times.
- The logical timescale for one or more space vehicles is coordinated direcly or indirectly from Earth.
- Other vehicles coordinate with these vehicles using the interleaved symmetric protcol over the Proximity-1 space data link.
- PITS uses the same state variables as NTP and has the same error detection and recovery mechanisms.

IEEE 1588 Precision Time Protocol (PTP)



- Ethernet NIC hardware captures a timestamp after the preamble and before the data separately for transmit and receive.
- In each round master sends Sync message at T_1; slave receives at T_2.
- In one-step variant T_1 is inserted just before the data in the Sync message; in two-step variant T_1 is sent later in a Follow_Up message.
- Slave sends Delay_Req message at T_3; master sends Delay_Resp message with T_4. Compute master offset θ and roundtrip delay δ
  - offset $\theta = ½[(T_2 – T_1) + (T_3 – T_4)]$, delay $\delta = (T_4 – T_1) – (T_3 – T_2)]$
- Note that IEEE 1588 packets have room for only one timestamp.

PTP interleaved mode

- The interleaved technique used in NTP could be used in PTP to send T_1 in the next Sync message.
- This avoids the need for the Follow_up message.
- As the delay is measured separately by each slave, a lost Sync message is easily found and discarded.

Further information

- NTP home page http://www.ntp.org
  - Current NTP Version 3 and 4 software and documentation
  - FAQ and links to other sources and interesting places
- David L. Mills home page http://www.eecis.udel.edu/~mills
  - Papers, reports and memoranda in PostScript and PDF formats
  - Briefings in HTML, PostScript, PowerPoint and PDF formats
  - Collaboration resources hardware, software and documentation
  - Songs, photo galleries and after-dinner speech scripts
- Udel FTP server: ftp://ftp.udel.edu/pub/ntp
  - Current NTP Version software, documentation and support
  - Collaboration resources and junkbox
- Related projects http://www.eecis.udel.edu/~mills/status.html
  - Current research project descriptions and briefings