A Basic Principle of PTP Time Synchronization

Duy-Ky Nguyen, PhD 2005-Aug-26

1. Introduction

The **Precision Time Protocol** (**PTP**) is a protocol defined in the IEEE1588 – 2002 standard that allows precise synchronization of networks. At any time on a network, there is only one master PTP clock to do time synchronizing many slave ones attached to the network. However, any slave PTP can become a master based on the best master clock algorithm (more accurate) and finite state machine (orderly transition) mentioned in the IEEE 1588-2002 standard. In this note, we present only the basic principle how PTP provides time synchronization.

There are 4 basic PTP timing messages associated with 4 timestamps and 6 timing operations as described below

1. A master PTP stamps the time upon transmitting a **Sync** message (*preciseOriginTimestamp*[1]);
2. The master then sends *preciseOriginTimestamp* in a **FollowUp** message;
3. A slave stamps the time upon receiving the **Sync** message (*sync_receipt_time*[2]);
4. The slave stamps the time upon sending a **DelayRequest** message (*delay_req_sending_time*[3]);
5. The master stamps the time upon receiving **DelayRequest** message (*delayReceiptTimestamp*[4]);
6. The master then sends *delayReceiptTimestamp* in a **DelayResponse** message

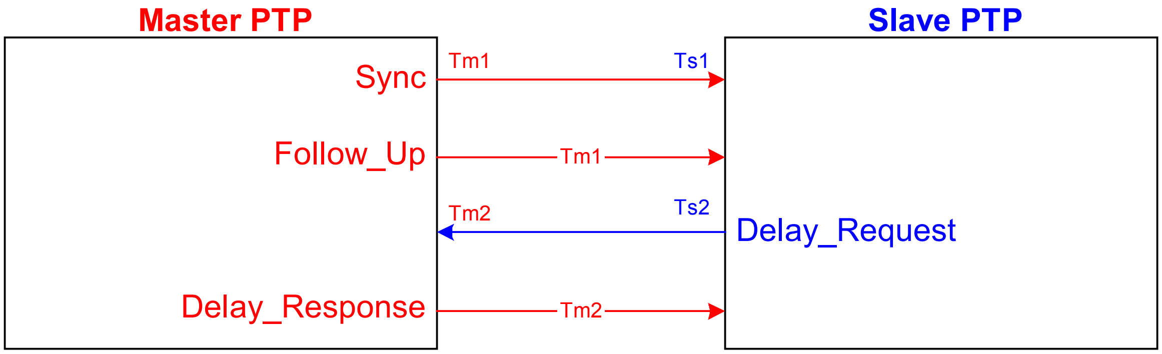
2. Computation

Practically we have a running master PTP clock and a slave one just powered up, so

$$T_m = T_s + T_o \tag{1}$$

where T_m, T_s are master and slave clock time, respectively; and T_o is offset time of slave PTP from the master. The goal is to sync a slave to a master PTP, in other word to get zero offset time, *i.e.* $T_o = 0$.

The slave PTP clock will do all computations for its clock synchronized to the master, *i.e.* to achieve $T_o = 0$.



For the direction from master to slave PTP using Sync message, we have

$$T_{s1} = T_{m1} - T_o + T_{m2s} \tag{2}$$

where

T_{s1} is slave timestamp upon receiving Sync message from master (sync_receipt_time);

T_{m1} is master timestamp upon transmitting Sync message in Follow-Up message (preciseOriginTimestamp after latency correction);

T_{m2s} is traveling time from master to slave.

For the reverse direction using Delay-Request, we have

$$T_{m2} = T_{s2} + T_o + T_{s2m} \tag{3}$$

where

T_{m2} is master timestamp upon receiving Delay-Request message from slave as in Delay-Response message (delayReceiptTimestamp);

T_{s2} is slave timestamp upon transmitting Delay-Request message (delay_req_sending_time);

T_{s2m} is traveling time from slave to master, i.e. in reverse direction.

Note opposite sign associated with T_o in Eqs. (2) & (3).

So, we have a system equation of only 2 equations (2) & (3) for 3 unknowns T_o, T_{m2s} and T_{s2m}. In algebra theory, a system equation can be solved only if number of *independent* equations and unknowns are equal. An independent equation cannot be derived from another one. We have used timestamp of Sync and Delay-Request messages for 2 equations above, they are independent. Timestamp of other messages Follow-Up and Delay-Response will produce *dependent* equations to Eq. (2) by adding some time T_x to both sides, like

$$\left(T_{m1} + T_x\right) = \left(T_{s1} + T_x\right) + T_o + T_{m2s} \tag{4}$$

Therefore, we have to assume symmetric transmission paths, so **one-way delay** is given by

$$T_d = T_{m2s} = T_{s2m} \tag{5}$$

and reduce to 2 unknowns T_o and T_d

$$\left.\begin{aligned} T_{s1} &= T_{m1} - T_o + T_d \\ T_{m2} &= T_{s2} + T_o + T_d \end{aligned}\right\} \Leftrightarrow \left.\begin{aligned} T_d - T_o &= \Delta_{m2s(S)} \\ T_d + T_o &= \Delta_{s2m(D)} \end{aligned}\right\} \tag{6}$$

where

$$\left.\begin{aligned} \Delta_{m2s(S)} &= T_{s1} - T_{m1} : master\_2\_slave\ delay\ (\text{Sync}) \\ \Delta_{s2m(D)} &= T_{m2} - T_{s2} : slave\_2\_master\ delay\ (\text{Delay Request}) \end{aligned}\right\} \tag{7}$$

Adding and subtracting Eq.(6), we have the solutions

$$\begin{cases} T_o = \dfrac{\Delta_{s2m(D)} - \Delta_{m2s(S)}}{2} \\ T_d = \dfrac{\Delta_{s2m(D)} + \Delta_{m2s(S)}}{2} \end{cases} \tag{8}$$

The slave PTP is synchronized to the master when $T_o = 0$, so, based on Eq.(1), the slave update clock given by

$$T_{s,update} = T_{s,current} + T_o \tag{9}$$

Note that Follow-Up and Delay-Response are solely used to send master timestamps to slave for computations.

3. Conclusion

Both master and slave do time-stamping on Sync and Delay-Request only. Slave PTP computes offset and one-way delay based on timestamps of its own and of master via Follow-Up and Delay-Response messages.