University of New Hampshire
InterOperability Laboratory

Clock Synchronization Terminology

by Jeff Laird, June 2012

Clocks

A **clock** is a device that measures either a point in time (time of day) or the passage of time (time interval). A clock comprises a stable **oscillator** and a **counter**. In most precise clocks the oscillator is either a quartz oscillator or an atomic oscillator.

Oscillators

> **A shift in fundamental oscillators -** Historically the most stable oscillator has been the earth rotating about its axis, so time of day and passage of time have both been based on the average **day**. A **second** has historically been defined as 1/86,400th of a day. If the earth's rotation speed changed slightly then the length of a second changed accordingly. In 1960 with the advent of more stable atomic oscillators the standard for the *passage of time* shifted to become the atomically defined **second**. *Time of day* is still based on the astronomically defined day. Interestingly, civil time (UTC) is defined in terms of the *second* not the *day*, so to keep civil time in sync with the astronomical day as the earth's rotation slows, a leap second is occasionally added to UTC.

Ceramic resonators are inexpensive but maintain a frequency tolerance of just 0.5% (20 s/hr, 7 min/day), so they are not used for precise clocks.

Crystal oscillators ("XO") (typically quartz) are the next step up, with a frequency tolerance of 0.001% (1 s/day, 5 min/yr). A 10°C temperature difference can make a 2 min/yr frequency difference, and a 20°C difference can make an 8 min/yr difference. Therefore the next steps up are temperature-compensated crystal oscillators (TCXO) and microcomputer-compensated crystal oscillators (MCXO). Beyond that are oven-controlled crystal oscillators (OCXO), which achieve the best frequency tolerance among crystal oscillators: 10^{-12} (1 s/30,000 years) over short periods of a few seconds, and 10^{-8} (1 s/3 years) over longer periods due to crystal aging.

Atomic oscillators are the best oscillators. Rubidium oscillators ("standards") are inexpensive ($500) and small enough to fit inside a computer as an adapter card. Their frequency tolerance is 10^{-9} (1 s/30 years). Cesium standards are expensive ($50,000). Their frequency tolerance is 10^{-12} (1 s/30,000 years).

An ideal system combines a local atomic clock for short-term stability with GPS corrections to compensate for long-term inaccuracy.

Clock Synchronization

Clock synchronization is a general term that includes

- **frequency synchronization**, a.k.a. **syntonization** - getting clocks running at the same rate
- **phase synchronization** - getting syntonized clocks in phase—"clapping in unison" so to speak
- **time synchronization** - getting clocks to agree on the time of day

Some clock synchronization applications require only *frequency synchronization*, for example circuit-based communication like telephone time-division multiplexing and synchronous Ethernet. Other clock synchronization applications require or are better suited for *time synchronization*. Packet-based communication channels, lacking a continuous frequency source, must use time synchronization even if the application needs only frequency synchronization. An interesting difference between circuit-based and packet-based synchronization is that traffic load affects the latter but not the former.

At its simplest, PTP is concerned with time synchronization between just two reliable clocks communicating only with each other over a reliable link. There is no aid from a third party such as a signaling clock equidistant from the two nodes.

> **Other clock synchronization research -** Much clock synchronization research involves multiple unreliable clocks in a fully connected network, but that is not relevant to the simpler PTP clock synchronization problem. In PTP, network-wide synchronization is achieved by establishing a master-slave hierarchy among the nodes so that each slave is involved in exactly one two-node synchronization process.

Network Delay

Syntonization can be achieved with *one-way communication* and no knowledge of network delay (assuming the network nodes are not moving). Phase synchronization and time synchronization, however, require knowledge of **one-way network delay**. Measuring one-way network delay without the aid of a third party requires *two-way communication*. If there is no delay asymmetry then the nodes can ascertain the one-way delay in each direction by measuring the round-trip delay and dividing by two. For this reason a key factor in these two types of clock synchronization is **network delay asymmetry**. Unknown asymmetry introduces a lower bound to clock synchronization. Perfect symmetry (or known asymmetry) allows arbitrarily precise synchronization, regardless of the delay.

> CAT5 cable asymmetry is 25-50 ns per 100 m (nist.gov/el/isd/ieee/upload/tutorial-basic.pdf p 76).

Unfortunately, there is no way to detect delay asymmetry without either involving a third party or first achieving time synchronization. Therefore, throughout clock synchronization research, network delay asymmetry appears as the primary uncertainty value that imposes a lower bound on the tightness of clock synchronization. With one-way delay uncertainty ε two clocks cannot be guaranteed to be synchronized any closer than ε.

Offset, Skew, and Drift

Section 10 of RFC 2330 *Framework for IP Performance Metrics* (and by reference RFC 1305) defines **clock offset** to be the clock's reported time T_c minus true time T_t, with true time in the world of IP being UTC. **Accuracy** is the absolute value of offset. By this definition a clock that gains two seconds per day can occasionally be accurate. For example, if it is set to be one second behind at the beginning of the day it will be accurate (accuracy = 0) at one point around midday. **Relative offset** is the offset between any two clocks. **Synchronization** is the minimization of relative offset, regardless of accuracy.

A clock's **frequency** is its cycles per true second. You might call it the first derivative of its "timestamp" (or "relative offset from a stopped clock") with respect to true time, just as speed is the first derivative of position. A clock's **skew** at time T is its frequency F_c at time T minus true frequency F_t. (One unconventional but possibly useful distinction might be to rename the "accuracy" defined above to "time accuracy", and then to introduce the term "frequency accuracy" defined as the absolute value of skew.) **Relative skew** is the skew between any two clocks. **Syntonization** is the minimization of relative skew (regardless of "frequency accuracy").

> In the next paragraph (and in the RFCs) the word "drift" is defined to mean something different than its normal English meaning. In normal English we say two stable clocks "drift apart" when they operate at different frequencies and so develop an increasing relative offset (technically "increasing accuracy", but that sounds like an improvement when the situation is actually degrading). According to the definitions in the RFCs we would more properly say those clocks "skew apart".

The RFCs do not define a single, general term for the second derivative of "timestamp" (or "relative offset from a stopped clock") with respect to true time, which is the same as the first derivative of frequency with respect to true time, i.e., the change in frequency through time. Rather, the RFCs define several special-case terms. **Jitter** is short-term variation in frequency, noticeable in frequencies above 10 Hz. **Wander** is longer-term variation in frequency, noticeable in frequencies below 10 Hz. The terms **drift** and **stability** are used for wander over long periods of time—from minutes to years. In clock synchronization (or at least in RFC 2330) "drift" is the preferred term.

As suggested above, these terms are closely related to *position*, *speed*, and *acceleration* in mechanics. An accurate clock has zero offset, which is like being in the right position. A clock operating at the proper frequency has zero skew, which is like traveling at the right speed (but possibly in the wrong position). A clock that maintains its frequency perfectly steadily (but possibly incorrectly) has zero drift, which is like traveling at a constant speed with no acceleration.