802.1Qbv: Feedback on Draft D0.0

Rodney Cummings

National Instruments

Current Text in 802.1Qbv D0.0

Event name	Event parameter	Event action
Start	UTC-time	If the time specified by the UTC-time parameter is in the future, then the Start event delays execution of the next gate event until the specified time. If the specified time is in the past, then this gate event has no effect and the next event in the list is exe-cuted.

<<Editor's Note: I have attempted to capture/interpret the discussion in York (May 2012) on this topic. The use of a "start" event tied to an absolute time (preferably in the future) seemed to me to be a simple way to allow a coordinated start-up of multiple devices in a sheduled network following configuration of the various gate event lists in each device, but this may prove to be too naiive a solution - comments please.

Truly Absolute Time is a Problem

- Cannot set a default for Start (non-volatile)
  - Automotive example: Future ignition time is unknown
- If a manager end-station sets Start in all bridges and end-stations at power up, we face a choice…
  - Far future time: Long delay until ready (e.g. ignition)
  - Near future time: Device is late, Start in past so ignored, events at wrong times, control system failure

802.1AS Hints to a Solution

9.5 ClockTargetClockGenerator interface

9.5.1 General

This interface is used by the ClockTarget entity to request that the ClockSlave entity deliver a periodic clock signal of specified period and phase. The ClockTarget entity invokes the ClockTargetClockGenerator.invoke function to request that the ClockSlave entity generate the periodic clock signal. The ClockSlave entity invokes the ClockTargetClockGenerator.result function at significant instants of the desired clock signal.

9.5.2 ClockTargetClockGenerator.invoke parameters

9.5.2.1 clockPeriod (TimeInterval)

The value of clockPeriod is the period between successive invocations of the ClockTargetClockGenerator.result function. A value that is zero or negative causes any existing periodic clock signal generated via this application interface to be terminated.

9.5.2.2 slaveTimeCallbackPhase (ExtendedTimestamp)

The value of slaveTimeCallbackPhase describes phase of the generated clock signal by specifying a point on the PTP timescale such that ClockTargetClockGenerator.result invocations occur at synchronized times that differ from slaveTimeCallbackPhase by $n \times$ clockPeriod, where n is an integer.

NOTE—The value of slaveTimeCallbackPhase can be earlier or later than the synchronized time the ClockTargetClockGenerator.invoke function is invoked; use of a slaveTimeCallbackPhase value in the future does not imply that the initiation of the periodic clock signal is suppressed until that synchronized time.

Suggested Change to 802.1Qbv Text

- Replace the term “Start” with “StartPhase” in relevant locations.
- Replace the text for “Event action” of “StartPhase” with
  - “Specifies a point on the PTP timescale such that the next gate event in the list delays execution until the next synchronized time that differs from StartPhase by *n* x *cycle*, where n is an integer, and *cycle* is the gating cycle (3.4).

    NOTE - When StartPhase is the first gate event in the list, the value of StartPhase can be earlier or later than the current synchronized time at system initialization; use of a StartPhase in the future does not imply that the execution of the next gate event is suppressed until that synchronized time.”

Additional Startup Issue

- With preceding change, statically configured bridges power up and execute schedule at correct phase
- End-stations cannot transmit scheduled traffic until all bridges execute the schedule (i.e. not as best-effort)
  - Protocols like FlexRay have mechanisms to detect this
  - Control applications (e.g. automotive OEMs) often have requirements from power up to this schedule-ready time
- Items to keep in mind…
  - ISIS-SPB mechanism for end station to detect that all relevant end-stations and bridges are running schedule
  - Power up time for schedule is important (including 802.1AS)

Relationship of Event List to Queue

- Relationship in draft D0.0 is one to one

An implementation may support a list of gate events associated with each queue that change the transmission gate state for that queue according to defined conditions.

- Implies a distinct list of gate events per queue
  - StartPhase and Repeat may not be same for all queues
  - Multiple schedulers required per port (one per queue)
- Limit to a single event list per port (all queues)?
  - Meets requirements for automotive/industrial
  - Easier to understand
  - Easier to implement (?)

Thank you