802.1 Time-Sensitive Networking (TSN) on 802.3cg Multidrop networks

AUTHOR: CRAIG GUNTHER, HARMAN INTERNATIONAL
SUPPORTERS: DON PANNELL, MARVELL
RODNEY CUMMINGS, NATIONAL INSTRUMENTS

September 2017

WHAT THIS PRESENTATION IS AND IS NOT

The intent of this presentation is to determine if a TSN-like environment _can_ be built on a CSMA/CD multidrop medium. It does not attempt to address if it _should_ be done. That is a decision for the 802.3cg Task Force.

A TMDA-like environment can be accomplished by using AS to synchronize time across all nodes, then using Qbv scheduling to eliminate collisions.

Two TSN standards will be presented:

- Time Synchronization (802.1AS a.k.a. AS); required for any TSN-based solution
- Scheduled Traffic (802.1Qbv a.k.a. Qbv); a possible solution for TDMA-like access on CSMA/CD multidrop. 802.1Qbv requires time synchronization to coordinate transmission schedules across devices and therefore relies on AS.

Specific clause references in IEEE Std 802.1AS-2011 and IEEE Std 802.1Qbv-2015 are included to aid others in researching these topics further.

Caveat: This presentation represents the opinion of the author and is not an official presentation from the 802.1 TSN Task Group.

WHAT MULTIDROP SOLUTIONS HAVE BEEN PROPOSED TO DATE?

EPON over copper (Has a proposal for this been presented?)

- Node-to-node communication must go through the Master node, which means slave-to-slave communication uses twice the bandwidth
- Does the EPON proposal include MPCP GATE & REGISTER_REQ messages used by AS (Clause 13, Annex F)?

PLCA (PHY-Level Collision Avoidance)

- Node-to-node communication is direct and all nodes must be PLCA nodes
- Efficient use of bandwidth with minimal COMMIT/YIELD signaling overhead per node
- Latency can have a lot of jitter, but does have a calculable min/max
- May need a new 802.3 Study Group to standardize this solution

TDMA over CSMA/CD (802.1AS + 802.1Qbv)

- Node-to-node communication is direct and all nodes must be TDMA nodes
- Unused time-slots are wasted
- Latency is consistent

What are our requirements and goals?

- Collision free transmissions?
- Variable size packets?
- Constant or variable transmission rates from end stations?
- Efficient use of bandwidth?
- Deterministic minimal latency?
- Ease of configuration?

I would propose that these goals could be a little bit conflicting...

For example constant size packets and transmission rates could result in efficient bandwidth utilization and deterministic minimal latency on schedule media. I would propose an 802.1Qbv schedule would be ideal for this criteria.

Contrast that to varying packet sizes and intermittent transmissions along with efficient bandwidth usage could imply larger latency variations. In this case a PLCA-like approach may be preferable. Note that I could envision a simple approach to this where the Master station would coordinate transmissions from slave stations all via software without the need for PHY tokens.

For the 802.1Qbv portion of this presentation I will focus on deterministic minimal latency with ease of configuration. In order to accomplish this we may need to put less emphasis on efficient bandwidth utilization.

A software-based PLCA window inside a Qbv scheduled window would be interesting…

Running 802.1AS (time synchronization) on multidrop

Running 802.1AS on multidrop (summary)

While AS was originally targeted to run on point-to-point Ethernet links due to AVB's original goals, it grew to support shared media on IEEE 802.11, IEEE 802.3 EPON and other coordinated shared media links. Running AS on multidrop will need to consider the following two 802.3cg multidrop characteristics:

Collisions

Based on David Brandt's Addendum to Discussion of Multidrop Access Methods we can be sure that 802.3bf timestamps are valid on TX & RX packets. Any collisions can be detected and discarded outside of AS and AS will never see the collisions. Collisions on the TX side are handled since the 802.3bf TX timestamp delivered to AS will be for the successful TX packet. Likewise, RX timestamps will only be sent to AS when the associated RX packet is successfully received.

AS observed result would be that the RX packet may be delivered a little later than expected (because of collisions on earlier copies of the RX packet). This appears to AS as nothing more than a delayed AS packet, which can also happen when other frames are ahead of AS frames in the transmitter's egress queue.

Single Response

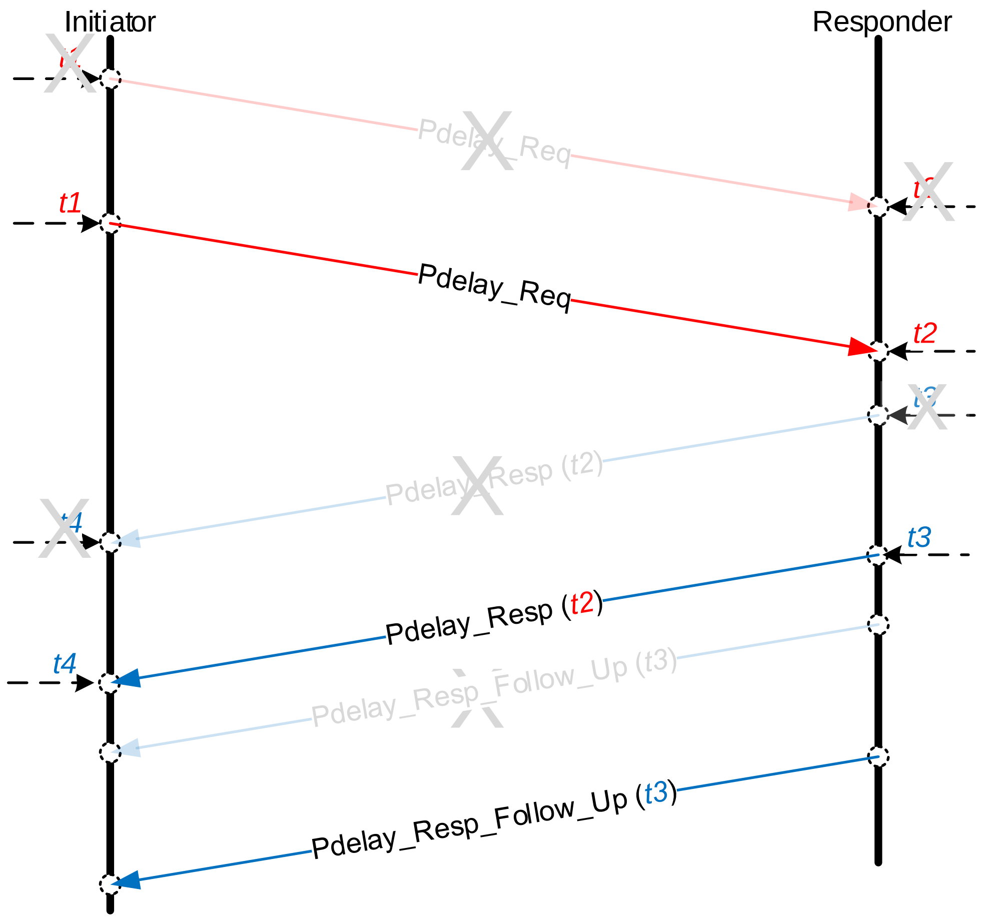
AS Messages use a multicast address 01-80-C2-00-00-0E (AS Tables 10-2 & 11-1), but only expect a single reply. If this multicast addressing is used on a multidrop network AS will be confused by multiple replies from all the other nodes on the shared media (AS 7.3.4, item b, 11.2.2). Therefore, a unicast addressing scheme (AS Annex E) must be implemented for some messaging.

802.1AS Propagation Delay collisions on multidrop

Diagram at right shows Pdelay exchange used to calculate propagation delay between nodes.

Propagation Delay formula (AS 11.1.2):

$$\frac{(t4 - t1) - (t3 - t2)}{2}$$

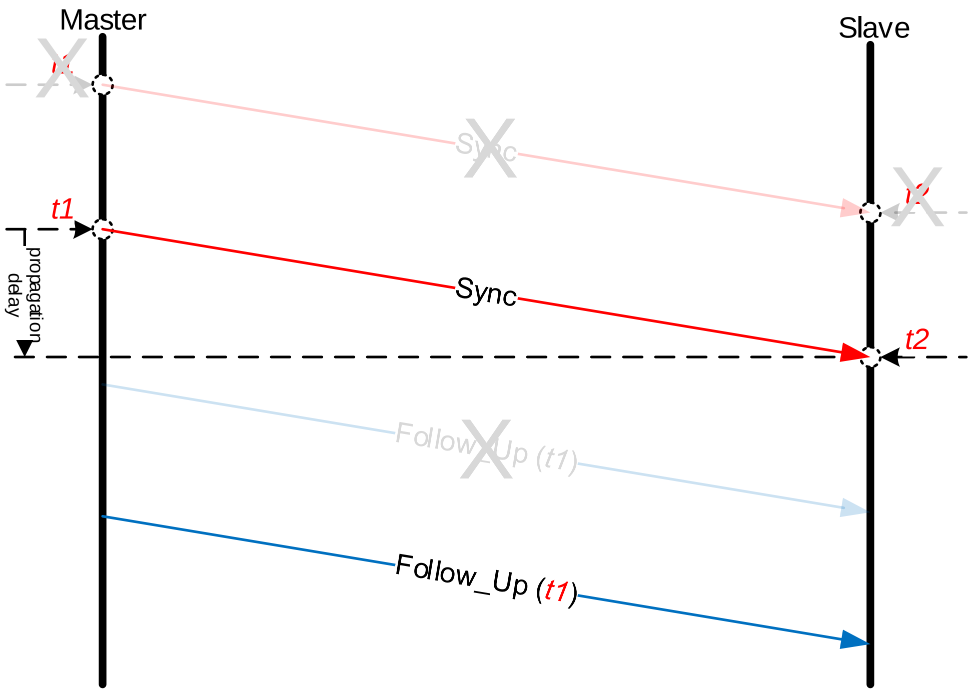


From the formula it is evident that the time between reception of the request (*t2*) and transmission of the response (*t3*) is irrelevant since that is subtracted from the time associated with the entire request/response transaction (*t4* - *t1*). Therefore, retries caused by collisions will not impact the Pdelay measurements as long as Response and Follow Up are received before the next Request is sent. Pdelay measurements occur once per second (AS 11.5.2.2).

Note 1: Cannot use layer 2 multicast addressing; that is discussed in a later slide.
Note 2: The Pdelay mechanism is also used to compute the ratio of the frequencies of the local clock to the peer's local clock to more accurately compute the propagation delay.

802.1AS Time Synchronization collisions on multidrop

Diagram at right shows Sync messages used to synchronize time.



Sync and Follow_Up message pairs are sent from the Master to the Slave (or to multiple Slaves as is the case in this multidrop proposal). In this situation it is okay, and actually desirable, to use the AS multicast address so that all slaves learn the current time at once.

There is no particular requirement for how soon the Follow_Up must be transmitted after the Sync as long as it occurs before the next Sync is sent, which is 8 times per second (AS 10.6.2.3, 11.5.2.3). The longer it takes the more "stale" the time is. Therefore, Syncs and Follow_Ups can communicate time in a collision environment.

Note: Follow_Up messages can also contain rate ratios and GM phase and frequency change information (AS 7.4).

802.1AS SINGLE RESPONSE ON MULTIDROP

Summarizing from the previous three slides:

1. Sync & Follow_up messages will use the layer 2 multicast address 01-80-C2-00-00-0E (AS Table 10-2 & 11-1). These messages are periodically sent from the Master to the Slave(s).
2. Announce messages should use the multicast address since there is no reply to these messages. Only the Master will periodically send this message.
3. Pdelay Request, Response, and Response Follow_up will use unicast addressing (AS Annex E). These messages are bidirectional between the Master and each Slave*.

The unicast addresses will be the MAC addresses of the devices in question.

ASSUMPTION: The Master will know the MAC addresses of all Slaves and all Slaves will know the MAC address of the Master.

* For shared media, we will need to run 802.1AS unicast, but there is plenty of precedent for that, so it is straightforward to include in an AS amendment.

How can unicast addresses be learned (not in AS - yet)?

IEEE 1588-Rev, Clauses 16.1,16.9 and 17.4 discuss various options with regard to unicast addressing. One technique to accomplish unicast configuration is described in 16.9 where the Announce message can contain a PORT_COMMUNICATION_CAPABILITIES or a PROTOCOL_ADDRESS TLV that tells other stations to communicate with this station via unicast addressing.

802.1AS ON MULTIDROP? YES

Can 802.1AS (gPTP) run on a multidrop network? Yes!

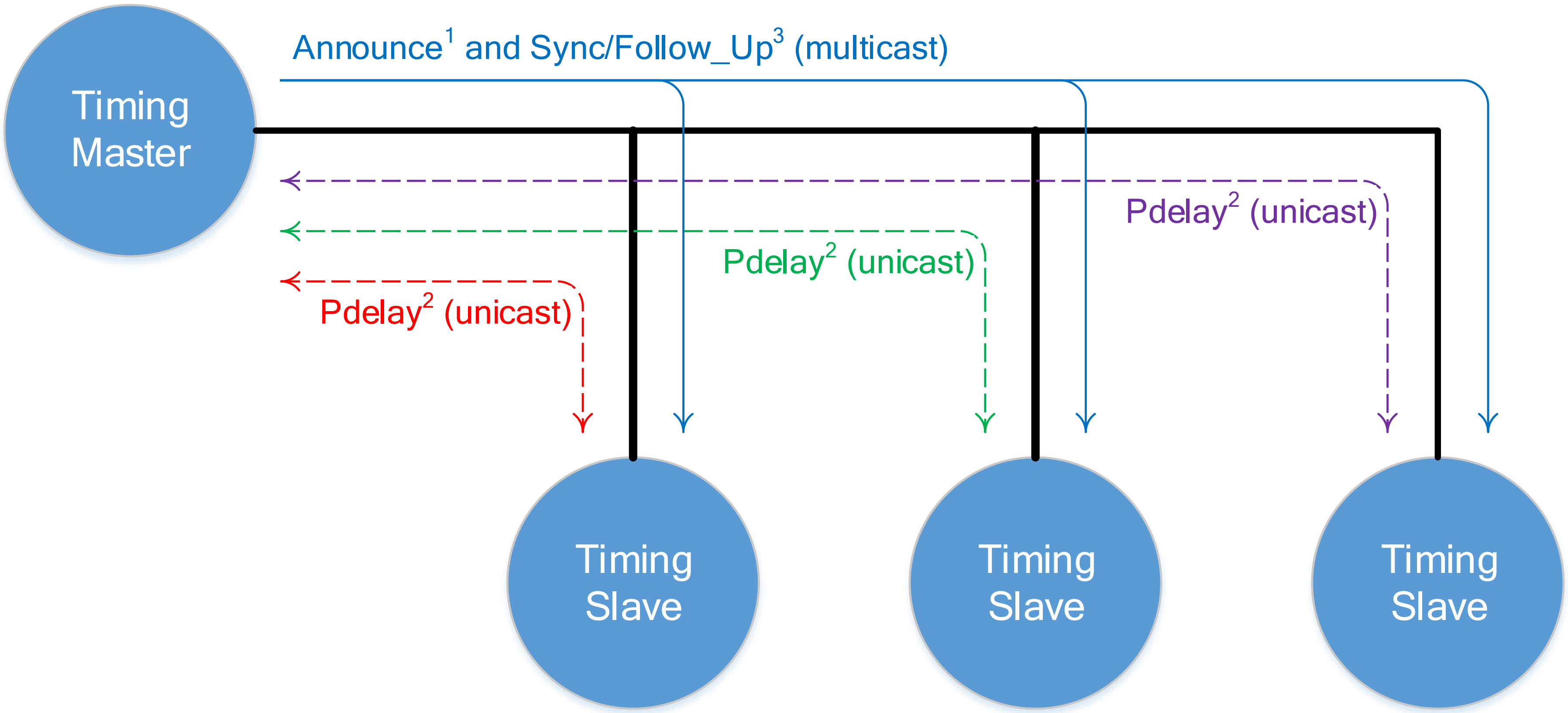
Here's the steps, assuming the timing Master for the multidrop segment will never change* (i.e. it is the port on the attached switch):

1. Master Announces itself to the network which allows Slave(s) to learn the Master's MAC address.
2. Slave(s) run Pdelay in unicast mode with the Master. Collisions will be handled appropriately. If the wire lengths are so short that the propagation delay is negligible it may be possible to skip Pdelay measurements in an engineered network. Slave(s) only need to track propagation delay to the single Master.
3. There is no need for the Master to run Pdelay propagation calculations against the Slave(s); therefore, Master does not need to track multiple delays.
4. Master transmits Sync & Follow_Up messages using the standard multicast address.
5. Slave(s) calculate current time by adding propagation delay (calculated by Pdelay) to the Master's gPTP time ('*t1*' from the Sync's Follow_up packet).

* Note: I believe the assumption about a 'dedicated' Master is not actually required if all nodes transmit Announce packets and the BMC algorithm chooses the Master node; the procedure described above will still work.

802.1AS ON MULTIDROP, PACKET EXCHANGES

The following diagram illustrates the packet exchanges detailed on the previous slide. In order to synchronize time, Slaves wait for (1) Announce which also contains the Timing Master MAC address, then run (2) Pdelay, then finally process (3) Sync/Follow_Up.



Running 802.1Qbv (scheduled traffic) on multidrop

RUNNING 802.1QBV ON MULTIDROP (SUMMARY)

An extract from the Qbv PAR states the scope of the amendment as:

> *enable bridges and end stations to schedule the transmission of frames based on timing derived from IEEE Std 802.1AS.*

Assuming it is important on an 802.3cg multidrop network to get deterministic low latency behavior with reduced delivery variation from shared CSMA/CD media, this portion of the proposal will address the following multidrop characteristic with regard to Qbv:

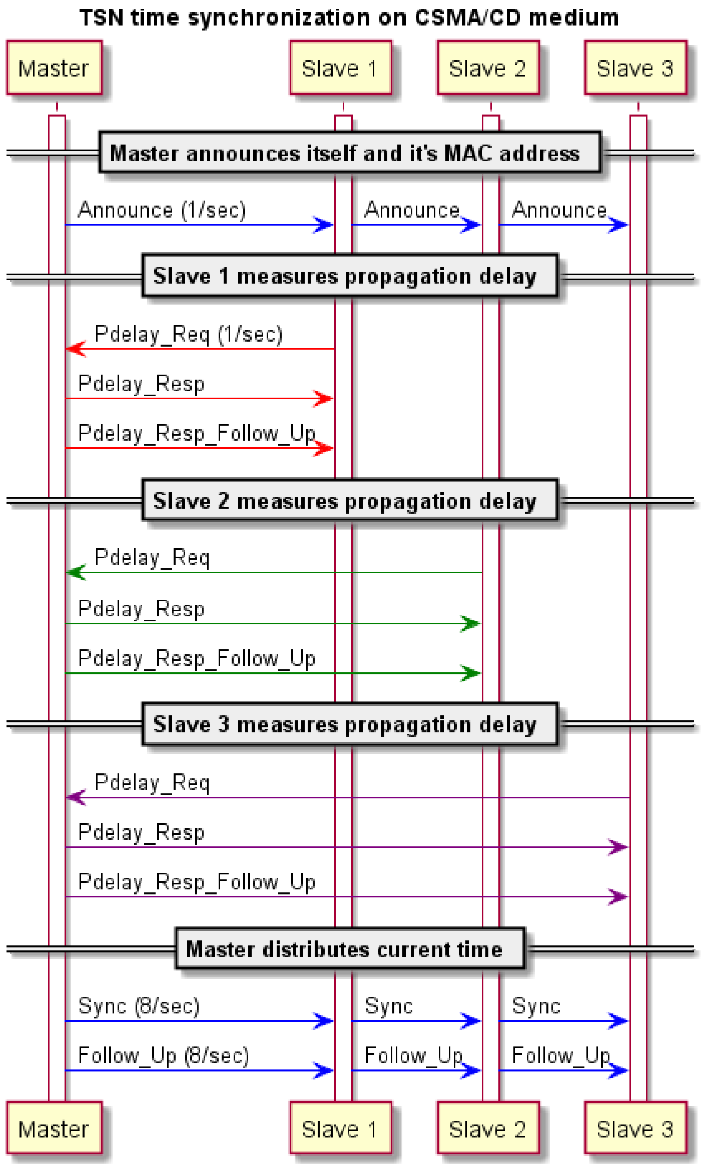
Collisions

Collisions cause retransmissions which increase latency and add an unknown amount of delivery variation to packet data. In order to solve both problems (increased latency and delivery variation) a TDMA approach can be implemented based on Qbv.

Qbv can be used to coordinate transmission of data among a group of devices. The transmission schedules created between these various devices, which are built on Qbv, are synchronized by timing information provided by AS. Note that Qbv was carefully written so that any PTP timing protocol, such as 1588, can be used; however, TSN focuses on AS (gPTP).

Once the Qbv schedule is established collision free traffic can begin flowing; this includes AS packets as well.

QBV SCHEDULE REQUIREMENTS FOR AS TRAFFIC @ 10 MBPS



	Announce	Sync	Follow_Up	Pdelay Req/ Resp/ Follow_Up
Size	90	64	90	68
FCS	4	4	4	4
Preamble+SFD	7+1	7+1	7+1	7+1
IFG	12	12	12	12
Total size	114	88	114	92
µsec xmit time	91.2	70.4	91.2	73.6

How many µsec does Master need per second?

Announce (91.2) +
3 * Pdelay_Resp & Resp_Follow_up (73.6+73.6) +
8 * Sync & Follow_Up (70.4+91.2)
= 1825.6 µsec

Each Slave needs:

Pdelay_Req (73.6)
= 73.6 µsec

Master + 3 Slaves take 1825.6 + (3 * 73.6) = 2046.4 µsec (~0.2%) of every second to keep time synchronized.

BUILDING A QBV SCHEDULE

Let's focus on the AS requirements:

Device	Packet	Rate (µsec/cycle)	Size	Duration (µsec)
Master	Announce	1,000,000	114	91.2
	Sync	125,000	88	70.4
	Follow_Up	125,000	114	91.2
	Pdelay_Resp (per slave)	1,000,000	92	73.6
	Pdelay_Resp_Follow_Up (per slave)	1,000,000	92	73.6
Slave x 3	Pdelay_Req	1,000,000	92	73.6

Conceptually, the simplest would be to open a window for each of the four devices once every 125,000 µsec, or 31,250 µsec per device. The downside is that each of the devices will not be allowed to transmit for 93,750 µsec while the other three devices go through each of their 31,250 µsec windows.

Too small of a window size would limit the maximum packet size, but allow more frequent transmit opportunities.

Simulating a “TDMA on 802.3cg multidrop”

The pre-engineered Qbv schedule for our simulation

The approach we plan to use in our simulations is to open windows big enough to allow a ~245-byte ping, which is 287+4+7+1+12 = 311 bytes (~248.8 μsec). We will use 250 μsec windows for each device to make it a nice even number.

Our schedule, which has a 1,000 μsec Qbv cycle time, will have four windows that open and close like this:

	t0	t0+250 μsec	t0+500 μsec	t0+750 μsec	t0+1000 μsec
Master	Open w1	Close w1			
Slave 1		Open w2	Close w2		
Slave 2			Open w3	Close w3	
Slave 3				Open w4	Close w4

Those familiar with 802.1Qbv will notice this is an unconventional use of Qbv, but completely contained within the specification. Also notice that this table shows adjacent windows closing and opening at the same instant; in reality there will be a small amount of time between closing a window and opening the next window.

How difficult will it be?

The simulation environment will be four 10 Mbps devices attached to a 10/100 hub.

In order to establish this TDMA approach in our devices we will need the following in our implementation:

1. Qbv schedule will be loaded, but not enabled, during boot.
2. AS will be implemented in the host controllers (see OpenAvnu for a good starting point) and modified as specified earlier in this presentation.
3. When AS time is synchronized the Qbv schedule will be enabled.

DONE! TDMA is operational and ready for testing!

Testing

Wireshark will be utilized to watch the traffic as the network boots. Our devices have an 802.1AS one PPS output we can use to watch for time synchronization. We may also be able to watch the Qbv cycles start. Tests will include 245-byte pings as soon as the network starts (i.e. collisions), and other tests waiting to start pings until after the Qbv schedule is enabled.

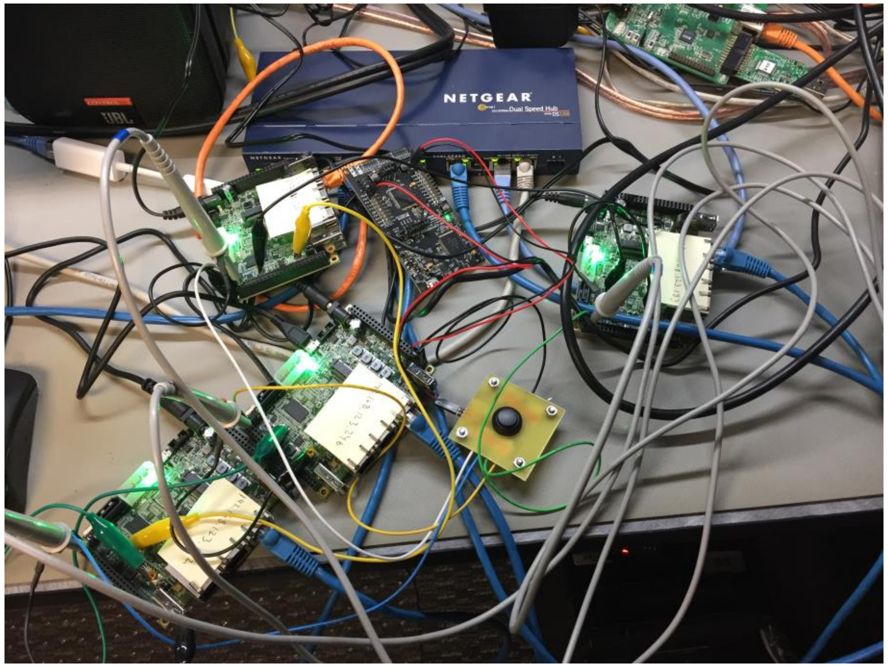
Results will be tabularized and reported when completed.

Last minute update… What we've done so far

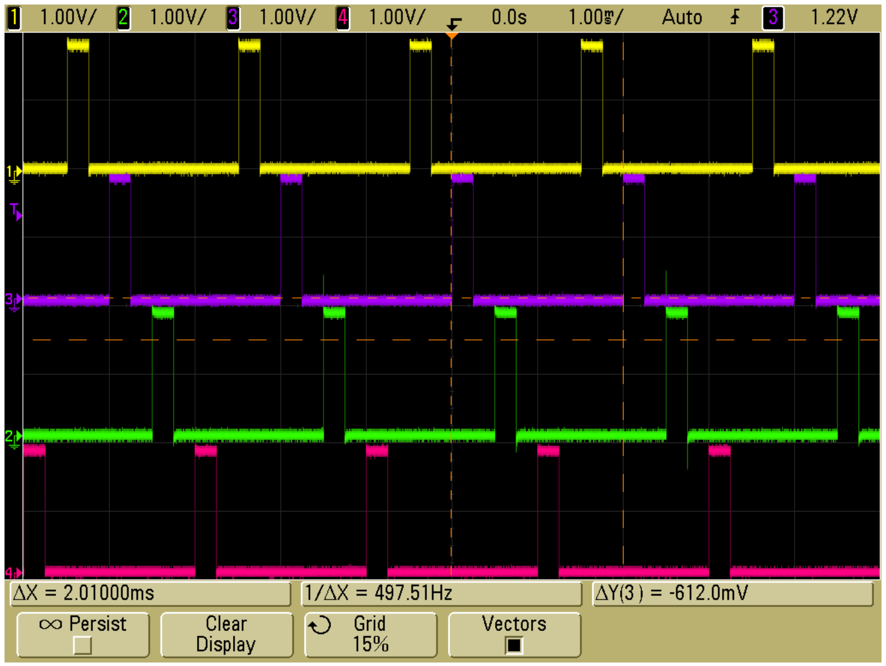
Currently we have 802.1Qbv running on four ESPRESSObin nodes connected to a Netgear DS108 hub. The nodes are configured at 10 Mbps.

Notice the big black pushbutton. We do not have 802.1AS (time synchronization) running yet, so we use the pushbutton (debounced by the other board you see) to synchronize gPTP time on the four nodes and then initiate the Qbv schedules.

Demonstration system:



Qbv schedule windows:



Post simulation

Once the simulation is complete and if there is sufficient interested in pursuing this approach, the modifications suggested to 802.1AS in this presentation will be submitted as part of a future project request for the TSN group.

It may be interesting to proceed with the 802.1AS changes even if the Qbv scheduled traffic technique for TDMA is not utilized. If 802.3cg develops a multidrop solution that requires time synchronization the 802.1AS changes presented in this presentation may still be required.

What’s next?

What can I do for the group going forward?

1. Is any of this interesting to this group?
   a) 802.1AS time synchronization
   b) 802.1Qbv scheduled traffic
   c) Software-based PLCA tokens
2. Should the TSN Task Group become more deeply involved?

TIME-SENSITIVE NETWORKING ON 802.3CG MULTIDROP

HARMAN

Questions? Discussion?

Thanks!