



Canova Tech

The Art of Silicon Sculpting

Dynamic PLCA Node ID Assignment

Nov, 4th 2020

- 802.3da has a formal objective to define an “optional PLCA node ID allocation method”
 - AKA “Dynamic PLCA”, or D-PLCA in short
- Three presentations given so far:
 - http://www.ieee802.org/3/SPMD/public/apr0820/spmd_nodeid_040820.pdf
 - https://www.ieee802.org/3/da/public/jul20/jones_spmd_01_0720.pdf
 - https://www.ieee802.org/3/da/public/102120/dalmia_3da_01_102120.pdf
- This presentation follows up providing new ideas to address concerns and questions expressed during the debate

- PLCA and TIS are defined in 802.3cg-2019 Clause 148 and 147
 - We have an objective to support interoperability with Clause 147
 - but PLCA belongs to Clause 148
 - This **implies** that we shall support Clause 148 interoperability in adding a method for PLCA node ID allocation
- Therefore, for introducing new features, I believe we should agree on a basic set of requirements first:
 - Do not make changes that would rule existing implementations non-compliant
 - Keep interoperability with non D-PLCA capable nodes as well as non PLCA capable nodes
 - Preserve the existing layering structure
 - Retain (or improve) existing performance

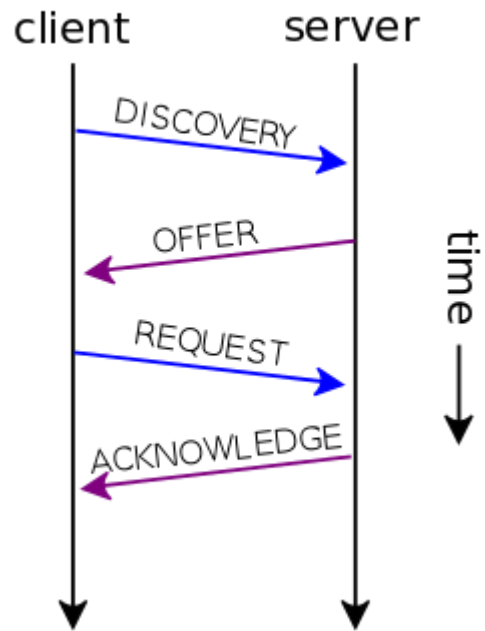
Where should the solution belong to?

- In principle, there are at least two different ways of solving the problem:
 - Define an upper layer method to set the relevant Clause 148 network parameters, without requiring PLCA to be configured upfront on a new node
 - Use an appropriate physical layer signaling to allow auto-configuration of the PLCA RS.
- The obvious advantage of the former is that it's flexible and it's inherently compatible with Clauses 147/148
- On the other hand, not requiring an upper-layer intervention to configure PLCA would reduce the overall system design effort and improve usability in some cases
- Which is best?
 - let's first explore what could possibly be achieved in the two scenarios
 - This presentation makes a proposal for each option, herein called the “upper layer solution” and the “physical layer solution”

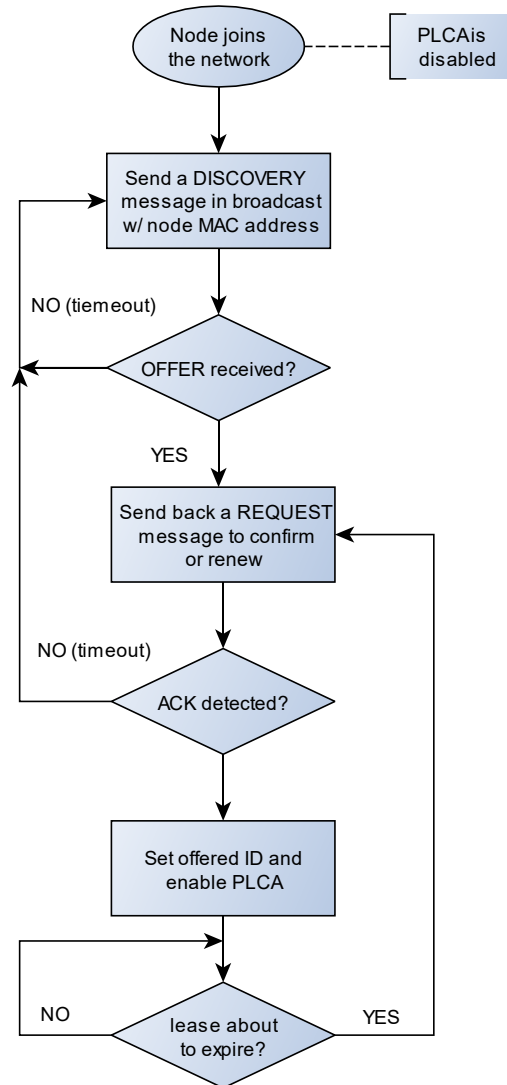
UPPER LAYER SOLUTION

- The problem is almost identical to the dynamic IP address assignment
 - DHCP (Dynamic Host Configuration Protocol) is a well-known and widely adopted solution which also supports redundancy, join/leave of nodes etc.
 - As R. Feynman used to say, “same equations, same solutions”
- “cg” nodes always start in “plain” CSMA/CD mode, using the Clause 22 RS (See the PLCA CONTROL and DATA state diagrams in Clause 148)
 - PLCA is in fact disabled by default
 - Nodes can ask for an ID using a DHCP-like protocol before enabling PLCA
- Should we really use DHCP?
 - Not necessarily, we could re-use the method and adopt a different protocol such as LLDP

Conceptual scheme



Example of protocol definition in the upper layer(s)



- We assume that each node is a client and there is at least one server on the network
- The client sends a DISCOVERY message to get enumerated
- 2-phases handshake to confirm
 - reject multiple offers, pick one
- Multiple servers can stay in sync by monitoring the OFFER/ACK on the line
- IDs are leased for a specific time
 - nodes should ask for renewal periodically
 - otherwise they are assumed to be absent
- The coordinator (ID=0) is selected likewise
- The server may need to change the `plcaNodeCount` on the node with ID=0
 - Can be done with a dedicated message

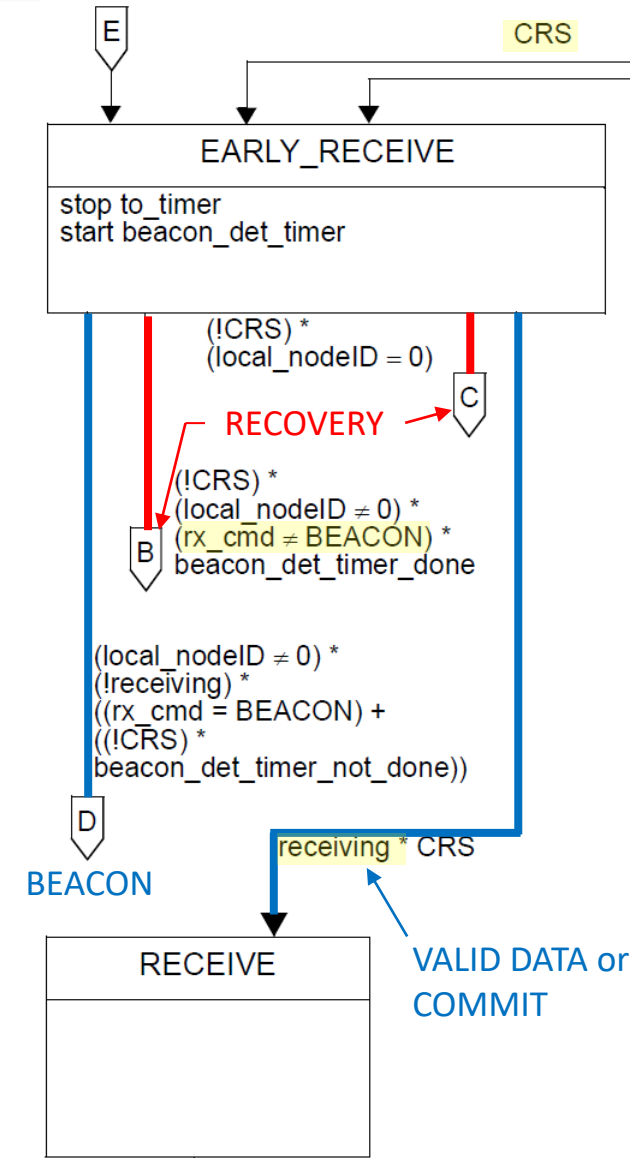
- Adding new nodes to a working PLCA enabled network will trigger collisions
 - not a problem in general but may be unwanted in specific applications
 - IDEA: the management entity can “reserve” TO #1 (the one just after the coordinator’s own TO) for enumeration.
 - Nodes awaiting for enumeration can enable PLCA already using node ID = 1
 - collisions are confined in TO #1 and will not affect already enumerated nodes
- Election of the coordinator (ID = 0) is not different from the enumeration of the follower nodes
- The actual policy for assigning which ID to which node (based e.g. on MAC address) would be application defined in the server

- The definition of the actual method to achieve enumeration is probably out of scope of 802.3
- What should be done in 802.3da then?
 - If using LLDP, define the appropriate TLVs
 - Ask 802.1 to define the actual method (?)
 - Ensure that the Physical Layer is providing all the necessary information via the management interface (Clause 30 and Clause 45)
 - We may consider adding a couple of status reports to C30 in addition to the existing ones, e.g.
 - Indication that a coordinator node is detecting “foreign” BEACONS (that is, BEACONS not generated by itself)
 - Indication that a node is not receiving transmit opportunities
 - Indication that a node is detecting “stolen” transmit opportunities (that is, duplicate IDs)
 - This may require very small additions to Clauses 148 and 30

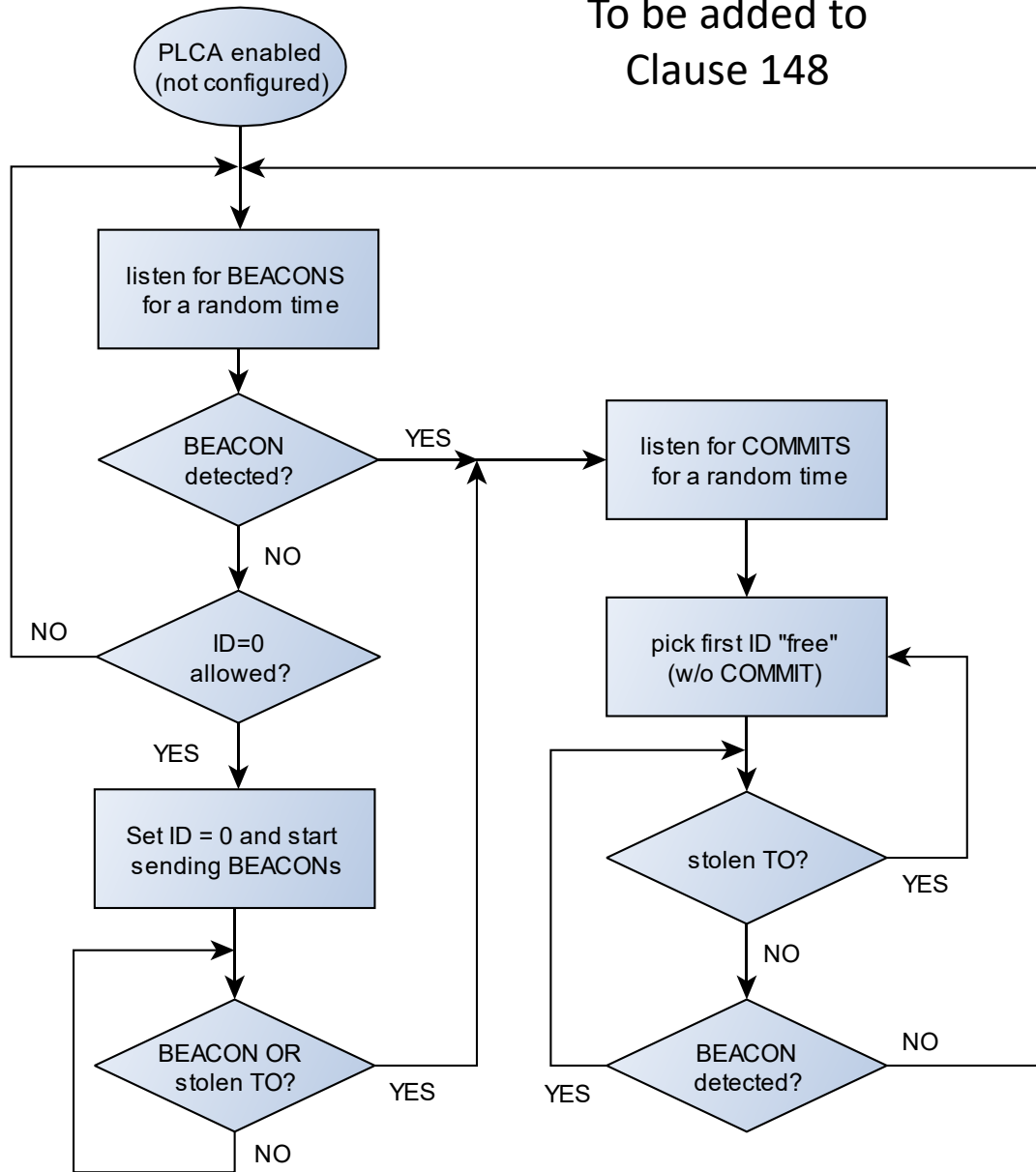
PHYSICAL LAYER SOLUTION

Constraints to preserve 802.3cg compatibility

- We shall not rely on handling detected collisions
 - Collision detection belongs to the Physical Layer but collision **handling** does not
- Any signal other than a valid preamble or COMMIT will be incompatible with Clause 148
 - That would make existing PLCA nodes go into a recovery state
- We should not define periodic transmissions on the line
 - PLCA nodes would react to that by signaling a collision in case of concurrent TX
 - performance penalty
 - non-PLCA enabled nodes will assert CRS at each transmission, causing deferral
 - may impact EMC/EMI performance
- That said, is it possible to design a physical layer solution fulfilling all these requirements?



To be added to
Clause 148

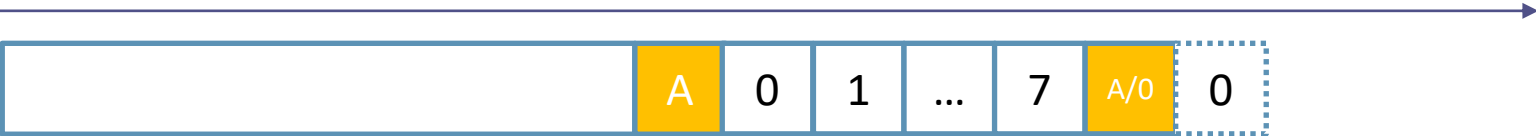


- Use the “duck” algorithm
 - “If it looks like a duck, swims like a duck, and quacks like a duck, then it probably is a duck“
 - Start over if it wasn’t
- Use the concept of “stolen TO”
 - detecting that some other node is transmitting during a node’s own TO using a COMMIT

- Elect a PLCA coordinator ($\text{localNodeID} = 0$) which sends BEACONS
 - nodes that are eligible for getting the coordinator role shall constantly monitor the line for BEACONS
 - if no BEACON is detected within some random time, set $\text{localNodeID} = 0$ and become the coordinator
 - if the coordinator detects a “foreign” BEACON or detects a COMMIT issued by another node within TO #0, relinquish the coordinator role and go for normal enumeration
 - **note** that this doesn't involve detecting collisions
 - Eventually, only one coordinator is selected by statistical convergence
 - Multiple BEACONS on the same mixing-segment affect performance/fairness of access but they don't prevent transmissions
- Have each node monitor the PLCA cycle continuously to collect a list of “occupied” TOs
 - This shall be done detecting COMMITS, **which also allows distinguishing non-PLCA nodes**
 - Pick the first “free” TO and set localNodeID accordingly
 - If at any time a node detects a COMMIT within its own TO, it shall relinquish the current ID and pick a new one
 - Mark a TO as “free” if no COMMITS are received within a specified time
- The coordinator node shall also dynamically adapt the plcaNodeCount parameter to the number of nodes detected
 - The plcaNodeCount sets the number of transmit opportunities between two BEACONS
 - Always keep at least one TO free ($\text{plcaNodeCount} > \text{highest ID detected}$), increasing plcaNodeCount accordingly
 - Decrease the plcaNodeCount if no node is claiming the TO before last (down to a minimum of 8)

Example: election of coordinator (localNodeID = 0)

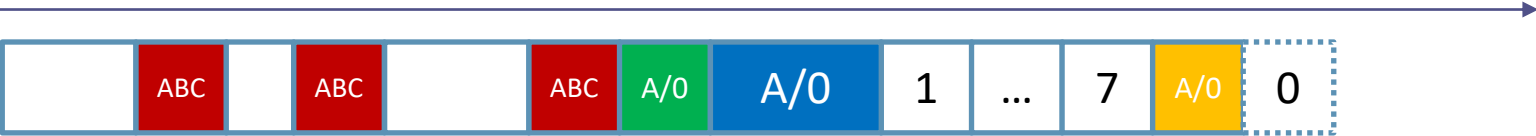
3 nodes (A, B, C) eligible to take the coordinator role, plcaNodeCount = 8 (default)








Ex 1: simple case, node A sends the BEACON first, nodes B and C “hear” it and renounce



Ex 2: BEACONS from A and B collide, then A and B hear the BEACON from C and both renounce.



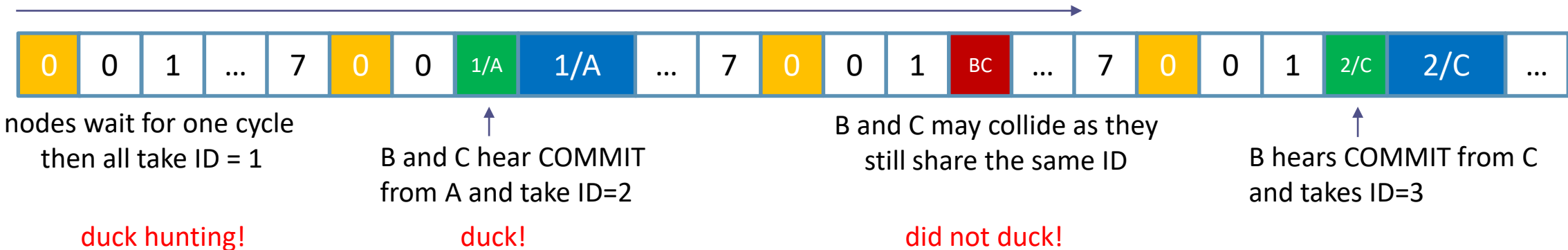
Ex. 3: worst case, BEACONS from A, B, C collide repeatedly then nodes B and C detect the COMMIT from A and renounce. If the packet from A collided, then **the MAC** would re-transmit after the usual random back-off (see also next slide)

-  BEACON
-  COLLISION
-  COMMIT
-  DATA
-  SILENCE

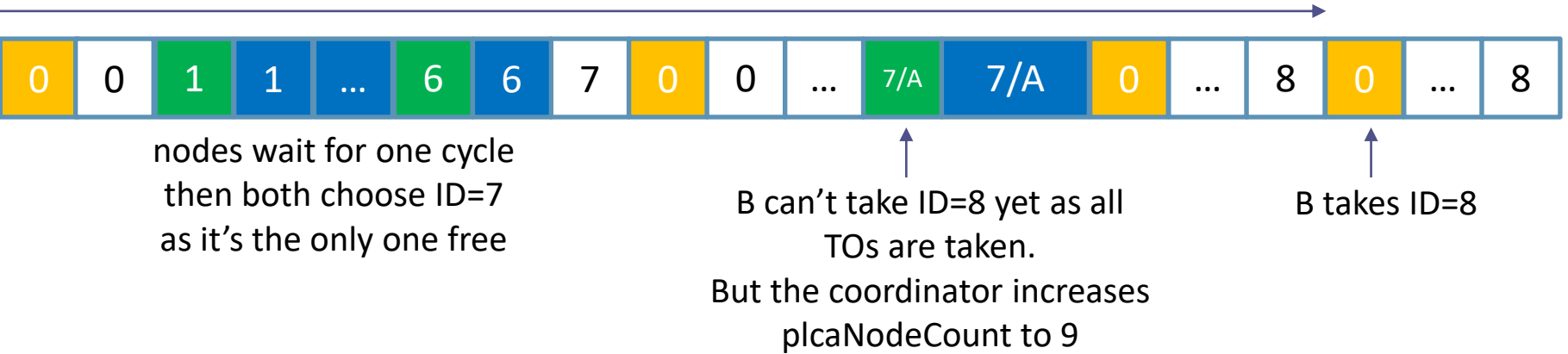
NOTE that during this time nodes can still send/receive data in plain CSMA/CD mode

Example: join of nodes

3 nodes (A, B, C) want to join, coordinator already selected, initial plcaNodeCount = 8




2 nodes (A, B) want to join, currently 7 on the network, initial plcaNodeCount = 8



- BEACON
- COLLISION
- COMMIT
- DATA
- SILENCE

- Relying on “normal” COMMIT signaling to resolve the ID conflicts may take a long time, depending on the network load
 - COMMITs are signaled by the PLCA RS after a node holding its transmission detects an incoming packet (i.e. a node with different ID took its TO).
 - a COMMIT is also issued when PLCA burst mode is enabled
- To speed up the enumeration process, we could either:
 - enable burst mode for a while when joining the network
 - no changes to Clause 148, but requires action from the management entity
 - modify Clause 148 to have D-PLCA enabled nodes always send a short COMMIT before (or at the end) of a packet, similar to burst mode
 - requires a small backward-compatible change to Clause 148

- How to deal with nodes leaving the network?
 - Since new nodes always take the first available TO they bet it's free, there is no need for other nodes to take any action, in principle.
 - However, the PLCA cycle length (i.e. the number of TOs set by `plcaNodeCount`) is dynamically adapted by the coordinator, as explained earlier
 - When `plcaNodeCount` is decreased, a node that remained silent for a long time may “lose” its TO as a result.
 - Possible solutions:
 - Have the follower node pick a new free TO when this happens
 - which may also result in the coordinator to increase `plcaNodeCount` again (not a problem...)
 - Never decrement `plcaNodeCount` and let the PLCA cycle adapt to the maximum number of nodes “seen” on the network at any time
-  • **Personally, I would not recommend following this path**

Mixing “cg” and “da” nodes

- What happens when mixing cg-compatible nodes with D-PLCA capable nodes?
- Case #1: plugging a D-PLCA node to an existing “cg” network
 - The D-PLCA node eventually works out a unique ID by listening on the spontaneous COMMITs sent by the “cg” nodes
 - In the meantime, it may create collisions which are properly detected by the “cg” nodes and handled by the MAC as normal
 - There may be no free TOs to take (i.e. the coordinator’s plcaNodeCount is equal to the actual number of nodes already)
 - In this case, the D-PLCA node won’t be able to achieve enumeration and will keep working in plain CSMA/CD mode creating random collisions.
 - This is what happens already if you plug a non-PLCA node to a PLCA network.
 - If the network load is low, the D-PLCA node may occasionally steal TOs from non D-PLCA nodes (not a problem...)
- 👉 — In no case the D-PLCA node can prevent a PLCA or non-PLCA node from transmitting, and vice-versa
- Case #2: plugging a “cg” node to a network having a D-PLCA capable coordinator
 - The coordinator will adapt to the highest ID configured in the “cg” nodes, if necessary
 - eventually, all D-PLCA nodes will detect the “cg” node presence by listening to COMMITs
 - The “cg” node will never release its ID (statically configured), but the D-PLCA nodes do!
- In short: the “cg” nodes win, the D-PLCA nodes adapt to them

CONCLUSIONS

- Two possible solutions have been presented to dynamically assign PLCA node IDs
 - Upper layer solution requiring from no to very little changes to Clauses 148
 - Physical layer solution requiring limited changed to Clause 148
 - does not break 802.3cg compatibility and fits into the Ethernet layering model
- Which one is better then?
 - Well, those are not mutually exclusive
 - As I often say, if you have to choose between “A” and “B”, pick “A and B”
 - I can see different applications benefitting from one or the other solution

THANK YOU

Constraints to preserve 802.3cg compatibility

- We shall not rely on detecting collisions
 - That would break layering. The **handling** of collisions lies within the MAC layer, while the **detection** and reporting belongs to the Physical Layer.
 - **NOTE:** the PLCA RS does not in fact handle collisions. Those are reported to the MAC via the PLS_SIGNAL primitive
 - Collision detection among very short transmissions is not reliable
 - It may be very difficult to distinguish a collision from noise in a short time window
 - The CSMA/CD protocol in Clause 4 mandates minimum slotTime and frameSize values for this purpose exactly
- Any signal other than a valid preamble or COMMIT will be incompatible with existing PLCA
 - That would make existing PLCA nodes go into a recovery state →
 - The rationale is that unrecognized, non collision-related carrier events on the line may indicate that the count of TOs within the PLCA control state diagram is wrong (see excerpt from Figure 148-4/b)
- We should not define periodic transmissions of COMMIT on the line
 - PLCA nodes would react to them signaling a collision in case of concurrent TX (performance penalty)
 - non-PLCA enabled nodes will assert CRS at each transmission, causing deferral
 - on loaded networks this may prevent a node from transmitting forever
 - may impact EMC/EMI performance
- That said, is it possible to design a physical layer solution fulfilling all these requirements?

