Specifying PLCA delay and overflow behavior

Wojciech Koczwara • | 8th May 2019

Rockwell Automation

Specifying PLCA delay and overflow behavior

Presenter: Wojciech Koczwara, Rockwell Automation

Presenter/Supporter: Piergiorgio Beruto, Canova Tech

Supporter: Tim Baggett, Microchip

1 Background

2 PLCA with small delay - no issues

3 PLCA with big delay (large number of nodes) - problem

4 PLCA with big delay (long to_timer) - problem

5 Comments

6 Proposed solution

7 Text changes

Background

Collision handling at MAC vs. slotTime

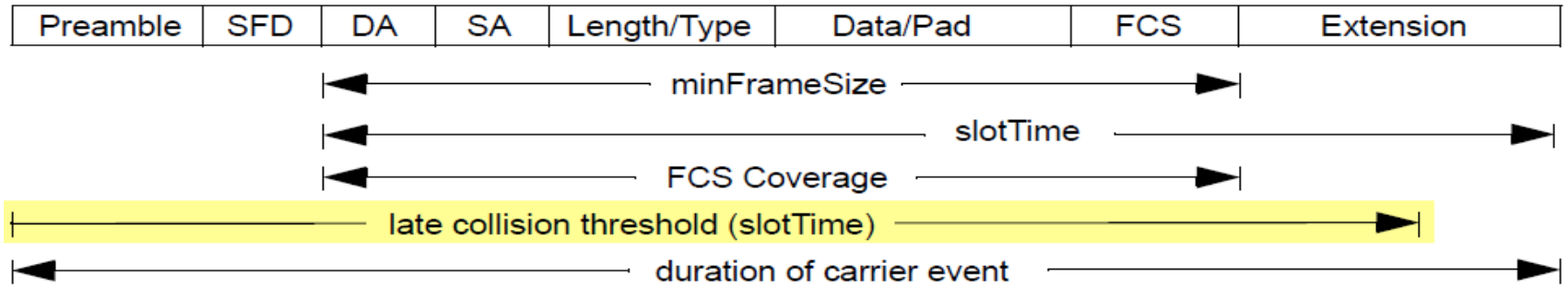


Figure 4–5—Frame with carrier extension

During sending, MAC reacts to PLS_SIGNAL.indication, reacting to the collisions and making new transmit attempts. This behavior is guaranteed until slotTime (512BT) has been reached during sending. After the slotTime has passed, MAC can report lateCollisionErrorStatus, and stop making further transmit attempts.

Late collisions should never happen on a properly functioning network.

Background

PLCA variable delay line

„The variable delay line is a small buffer that aligns a transmission with the transmit opportunity. The variable delay line length is no greater than to_timer × plca_node_count + beacon_timer.”

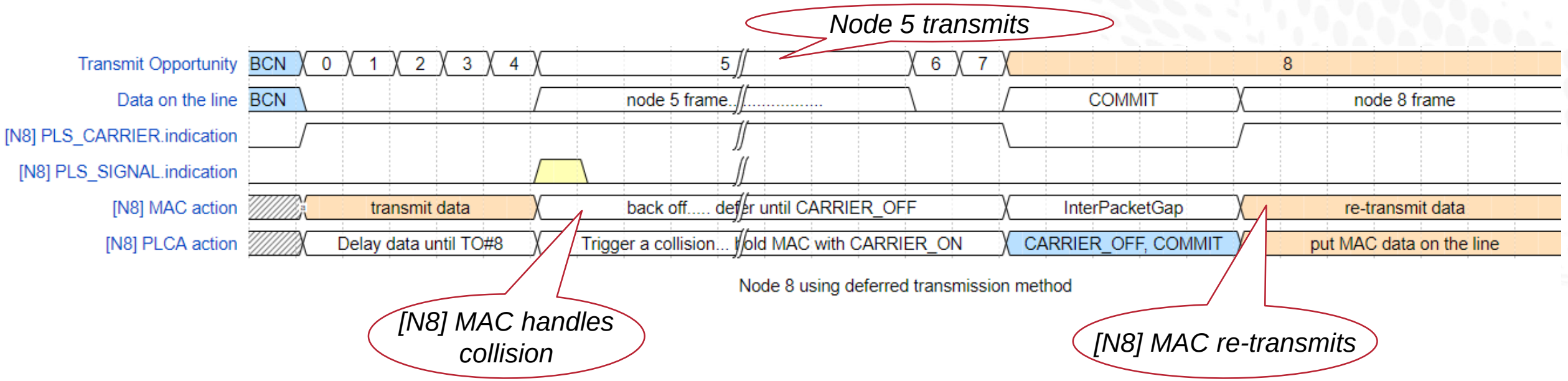
IF to_timer × plca_node_count + beacon_timer > slotTime

THEN the maximum delay line length can exceed the late collision threshold in MAC.

PLCA with small delay line – NO ISSUES

MAC of Node 8 starts sending, but node 5 sends data and delays Transmit Opportunity 8.

- When MAC of node 8 starts sending, PLCA uses the delay line to align transmission to Transmit Opportunity 8.
- If node 5 starts transmitting meanwhile, Transmit Opportunity 8 will come later than expected
- To limit the delay, node 8 PLCA asserts PLS_SIGNAL.indication to the MAC.
  - Node 8 MAC will back off for 0 or 512 BT, then make a new transmit attempt
  - PLCA uses PLS_CARRIER.indication to defer MAC re-transmission until Transmit Opportunity 8
  - At TO#8, PLS_CARRIER.indication is de-asserted. MAC defers for InterPacketGap, this time is filled with COMMIT, then MAC sends data which is put directly on the line.

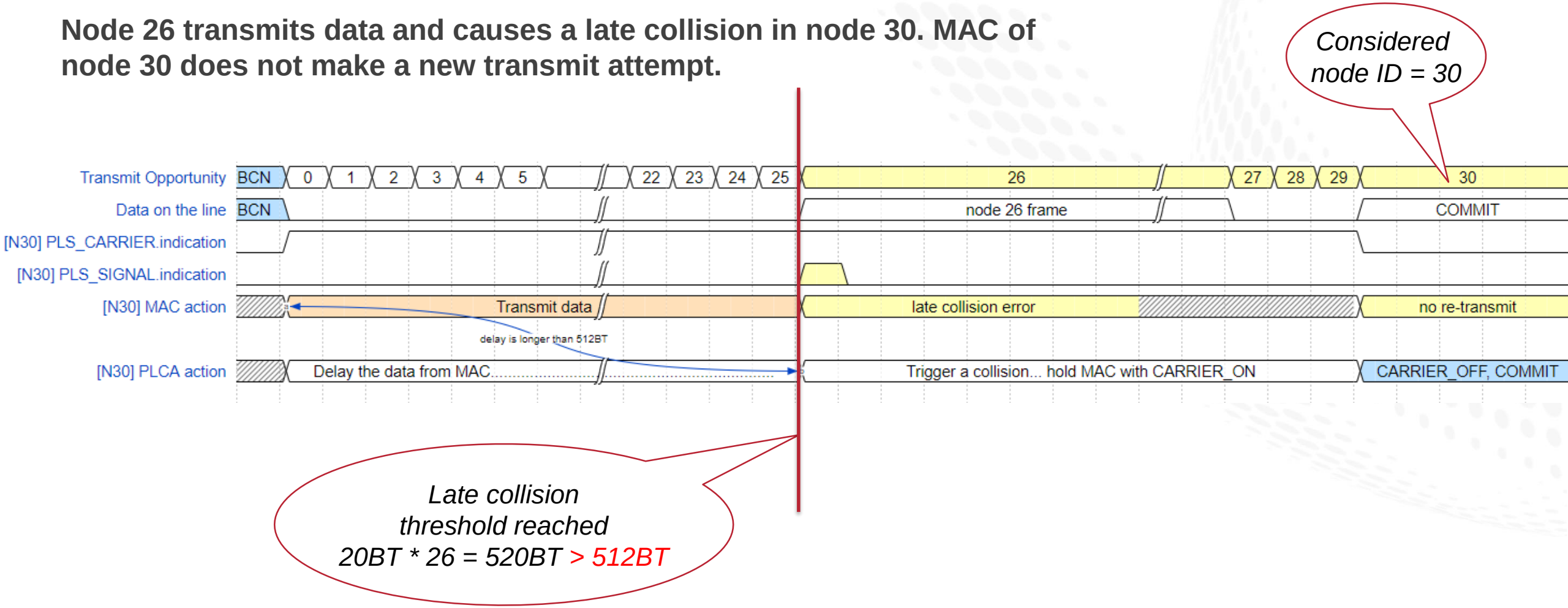


PLCA with big delay line (large number of nodes) - PROBLEM

Perspective of node 30, yellow fields highlight the problem

Due to excessive delay, late collision threshold (>=512BT) is reached in MAC.

Node 26 transmits data and causes a late collision in node 30. MAC of node 30 does not make a new transmit attempt.

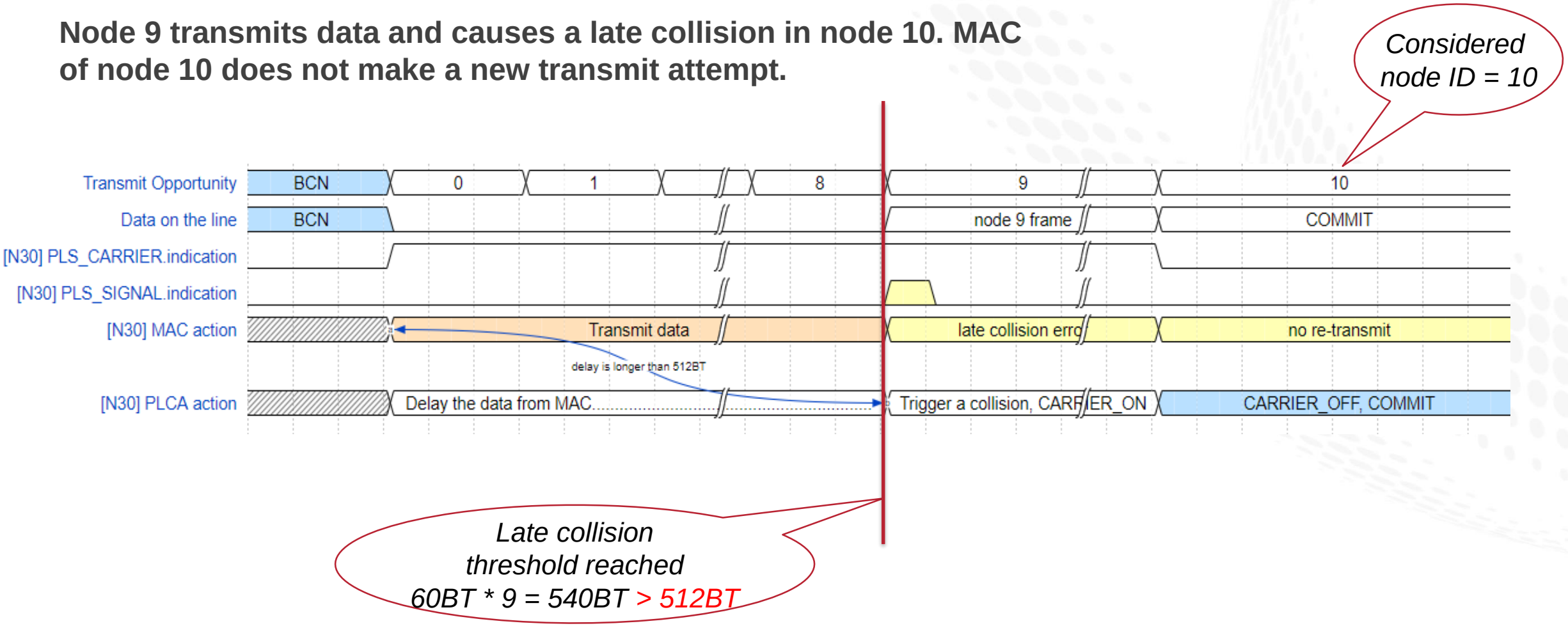


PLCA with big delay line (to_timer = 60BT) – PROBLEM

Perspective of node 10, yellow fields highlight the problem

Due to excessive delay, late collision threshold (>=512BT) is reached in MAC.

Node 9 transmits data and causes a late collision in node 10. MAC of node 10 does not make a new transmit attempt.



Comments

- i-427:

Even when the variable delay line length is less than slotTime, it is possible to configure a node to overrun the delay line before a transmit opportunity arrives. For example, if to_timer is set to 255 and there are more than 2 nodes, the delay line can fill before the transmit opportunity arrives. Other combinations of settings can lead to the same error.

- i-425:

*The existing draft allows configuration of compliant implementations in a way that violates a rule of CSMA/CD physical layer design - that the delay in the physical layer should not be allowed to be so long that late collisions can occur. The variable delay line length is allowed to be up to to_timer * plca_node_count + beacon_timer. The delay line should be limited to less than the slotTime in order to avoid late collisions.*

- i-198:

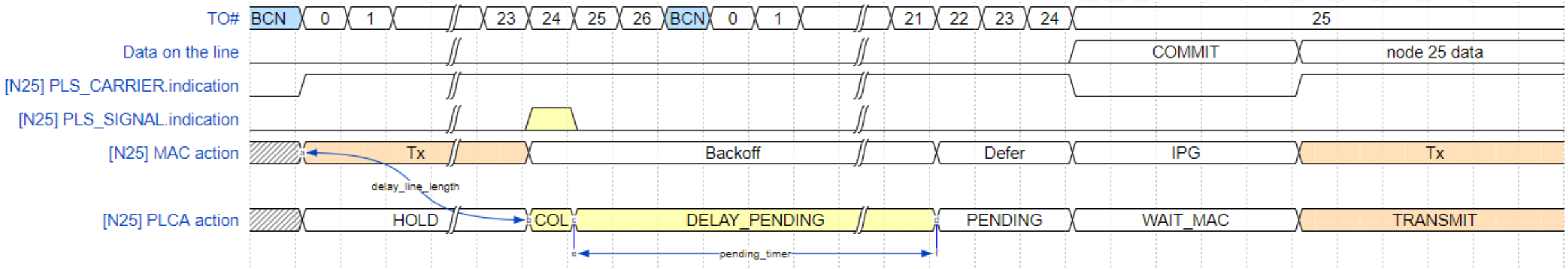
Variable delay line in PLCA RS can overrun slotTime, resulting in late collisions.

Proposed solution

The variable delay line length shall be limited to assure there are no late collisions.

When the maximum delay line length is reached, trigger a collision to the MAC.

Use pending_timer to cover MAC back-off time, then continue with the deferred transmission method.



Collide and defer transmission when the maximum length of variable delay line is reached.

Specifying PLCA delay and overflow behavior

Text Changes

V2.1

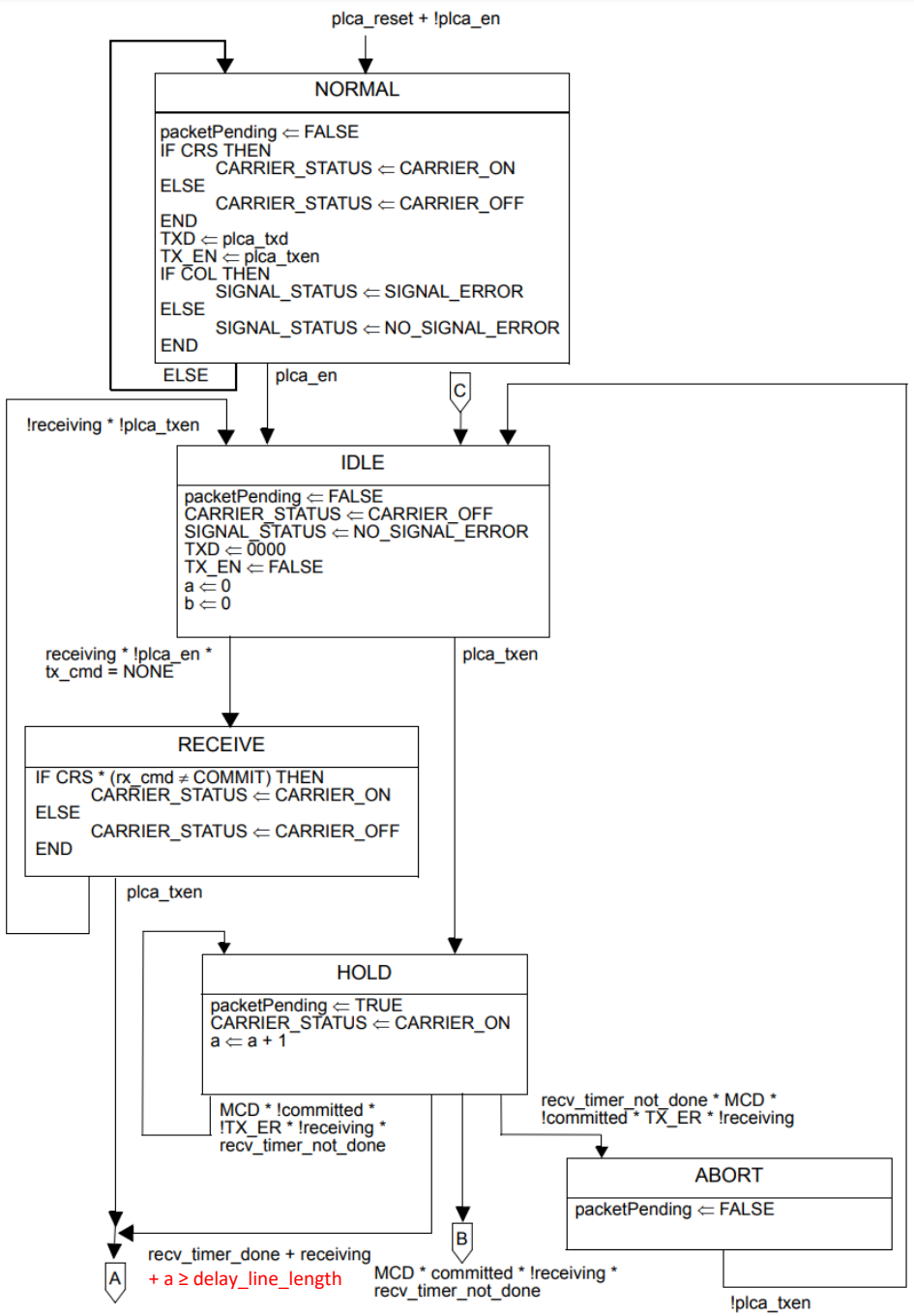


Figure 148–4—PLCA DATA state diagram

In Figure 148-4, in the transition from the HOLD state to the A connector, change the condition to:

“

recv_timer_done +
receiving +
a ≥ delay_line_length

“

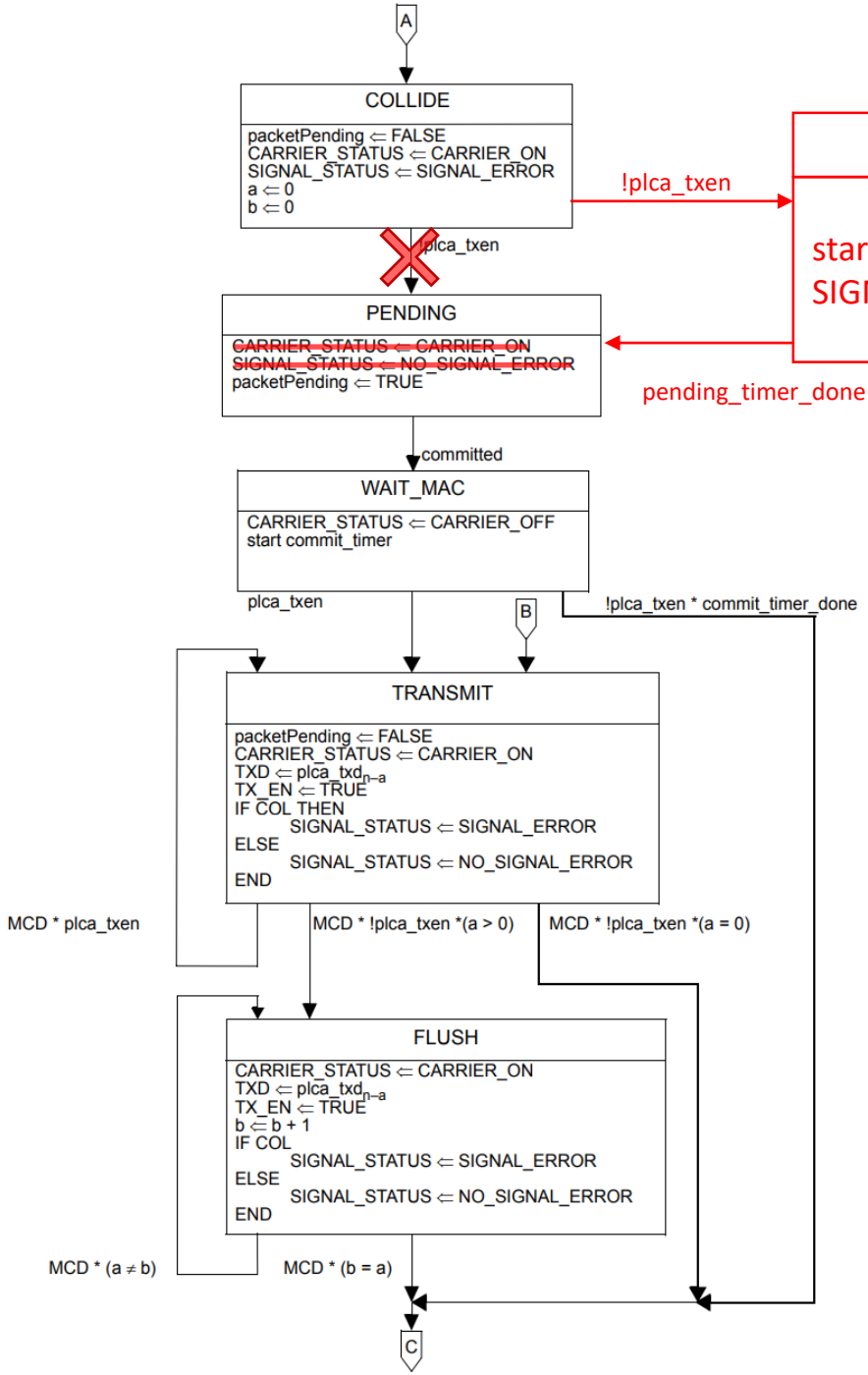


Figure 148–4—PLCA DATA state diagram *(continued)*

In Figure 148-4 do the following:

1. remove the transition from the COLLIDE to the PENDING STATE and its associated condition

2. In Figure 148-4, add a new state DELAY_PENDING between COLLIDE and PENDING states.

3. Add a transition between COLLIDE and DELAY_PENDING states with the following condition: “!plca_txen”

4. Add a transition between DELAY_PENDING and PENDING states with the following condition: “pending_timer_done”

5. Add the following text inside the DELAY_PENDING state box:
“start pending_timer
SIGNAL_STATUS <= NO_SIGNAL_ERROR”

6. From the PENDING state delete “CARRIER_STATUS <= CARRIER_ON” and “SIGNAL_STATUS <= NO_SIGNAL_ERROR”

Grant editorial license to draw the diagram according to IEEE 802.3 style

148.4.6.1 PLCA Data State Diagram

The variable delay line is a small buffer that aligns a transmission with the transmit opportunity. ~~The variable delay line length is no greater than to_timer × plca_node_count + beacon_timer.~~

[…]

During the COLLIDE state, the PLCA Data state diagram asserts packetPending = FALSE and CARRIER_STATUS = CARRIER_ON via the PLS_CARRIER.indication primitive. When the MAC is done sending the jam bits as described in Clause 4, it waits for the next transmit opportunity by switching to DELAY_PENDING state. The PLCA Data State Diagram switches to the PENDING state after waiting for the pending_timer. The pending_timer is used to prevent committing to a transmit opportunity before transmit data is available. This prevents conveying unwanted long COMMIT requests to the PHY.

Append text to 148.4.6.4 Timers

pending_timer

Defined the time the PLCA Data State Diagram waits in the DELAY_PENDING state before switching to PENDING state. Duration: 512 bit times.

add subclause 148.4.6.5 Constants

delay_line_length

This constant is implementation dependent and specifies the maximum length of the PLCA RS variable delay line depicted in figure 148-2.
Value: up to 480 bit times

148.4.3.1.2 Semantic of the service primitive

PLS_DATA.request (OUTPUT_UNIT)

The OUTPUT_UNIT parameter can take one of three values: ONE, ZERO, or DATA_COMPLETE. It represents a single data bit. The values ONE and ZERO are conveyed by the PLCA variables plca_txd<3>, plca_txd<2>, plca_txd<1>, and plca_txd<0>, each of which conveys one bit of data while plca_txen is set to TRUE. The value DATA_COMPLETE is conveyed by setting the variable plca_txen to FALSE. MII signals TXD and TX_EN are generated by way of the PLCA DATA state diagrams specified in 148.4.6. Synchronization between the RS and the PHY is achieved by way of the TX_CLK signal.
The mapping of this primitive to the plca_txen and plca_txd variables shall be accomplished in less than or equal to 8 bit times.

Thank you

www.rockwellautomation.com

Rockwell Automation