



Canova Tech

*The Art of Silicon Sculpting*

**PIERGIORGIO BERUTO**  
**ANTONIO ORZELLI**

*IEEE802.3cg TF*

*Comments on draft*

*May 18<sup>th</sup> 2018*

Error occurred when porting the picture to Frame from draft 1.0

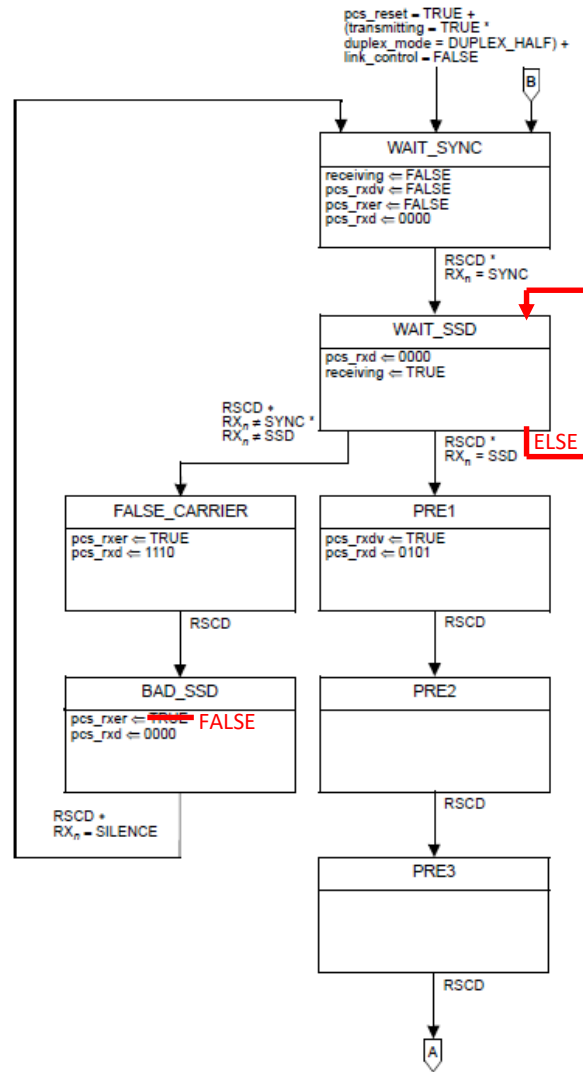


Figure 147-6—PCS Receive state diagram (1 of 2)

# #317, #318 → Scrambler - Proposed text changes

- **147.1.2 Operation of 10BASE-T1S**
  - The 10BASE-T1S PHY utilizes two level Differential Manchester Encoding (DME) modulation transmitted at a 12.5 MBd rate ( $\pm$  TBD). **A 17-bit self-synchronizing scrambler is used to improve the EMC performance.** 4B/5B encoding is used to further improve EMC performance and to perform out-of-band signaling among the connected PHYs. [...]
  - The 4B/5B mapping **is and the scrambler are** contained in the PCS (see 147.3) while the DME encoder/decoder is contained in the PMA (see 147.5).

# #319 → Scrambler - Proposed text changes

---

- **147.3.2.3 Functions**

- ENCODE

In the PCS transmit process, this function takes as its arguments ~~the pcs\_txd input~~ one data nibble, scrambles it as defined in 147.3.2.5 and returns the corresponding 5B symbol as defined in Table 147–1.

# #320 → Scrambler - Proposed text changes

- 147.3.2.5 Self-synchronizing scrambler

The PCS Transmit function shall implement multiplicative scrambling using the following generator polynomial:

$$g(x) = 1 + x^{14} + x^{17}$$

An implementation of self-synchronizing scrambler by linear-feedback shift register is shown in figure TBD#1. The bits stored in the shift register delay line at time  $n$  are denoted by  $Scr_n[16:0]$ . At every MII clock cycle, for each bit of  $TXD[3:0]$  the scrambler is advanced by one bit, and the output bit  $Sd_n[i]$  represented by the exclusive OR of  $Scr_n[13]$ ,  $Scr_n[16]$  and  $TXD[i]$  is shifted in as new  $Scr_n[0]$ , with  $i$  ranging from 0 to 3 (i.e. LSB first). The scrambler is reset upon execution of the PCS Reset function. If PCS Reset is executed, all bits of the 17-bit vector representing the self-synchronizing scrambler state are arbitrarily set. The initialization of the scrambler state is left to the implementer. In no case shall the scrambler state be initialized to all zeros.

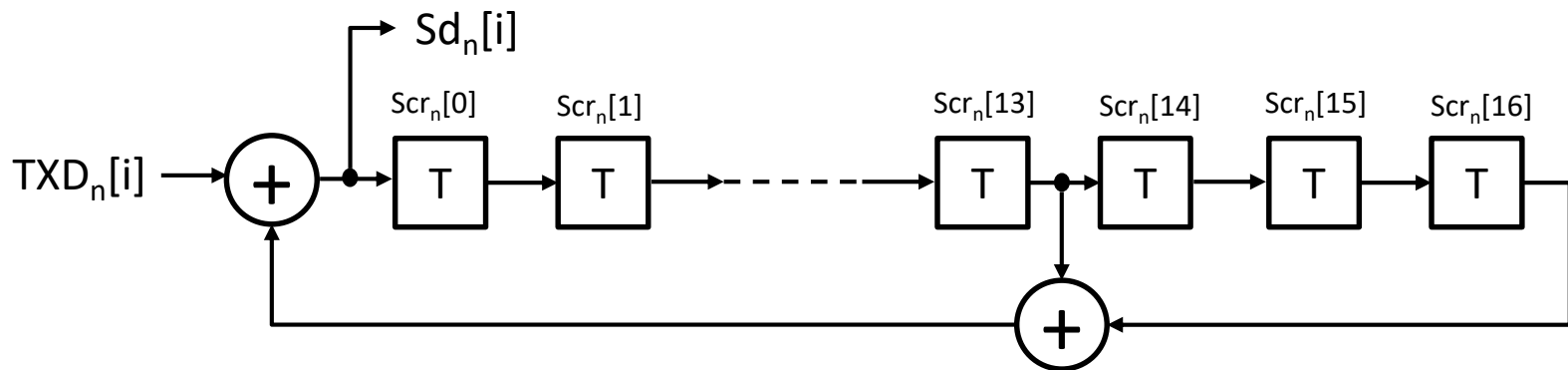


Figure TBD#1

# #321 → Scrambler - Proposed text changes

- **147.3.3.2 Functions**

- DECODE

In the PCS Receive process, this function takes as its arguments ~~the sym\_rx input data from PMA~~ one 5B symbol, decodes the corresponding nibble as defined in Table 147-1, descrambles it as defined in 147.3.3.4, and returns the corresponding **4B** MII data ~~nibble as defined in Table 147-1~~. If a violation of the encoding rules is detected, PCS Receive asserts the signal RX\_ER for at least one symbol period.

# #322 → Scrambler - Proposed text changes

- **147.3.3.1 Variables**

duplex\_mode      [...]

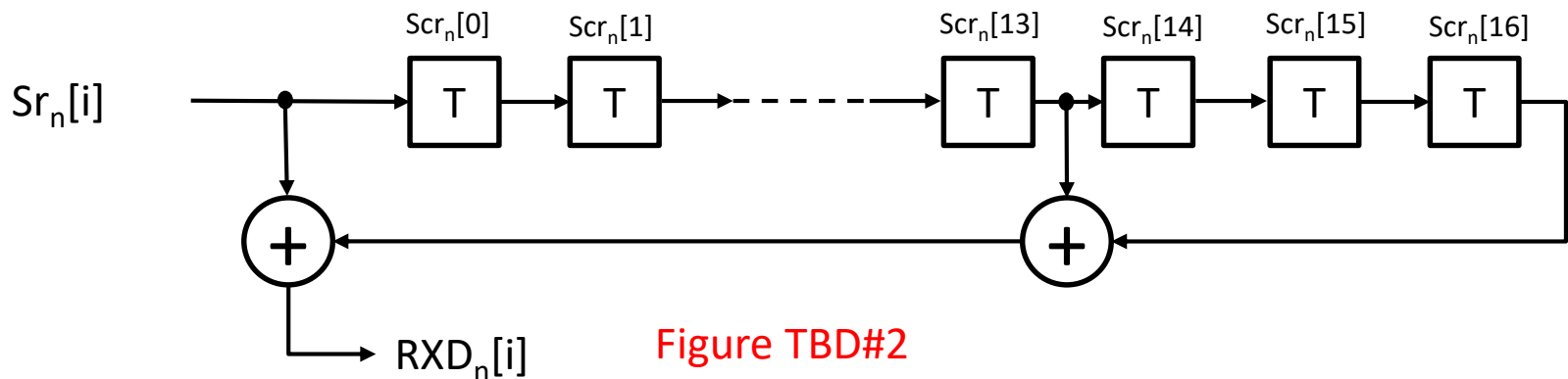
**precnt**                      **counter for preamble regeneration**

pcs\_rxdv              [...]

# #323 → Scrambler - Proposed text changes

## • 147.3.3.4 Self-synchronizing descrambler

The PCS Receive function shall descramble the 5B4B decoded data stream and return the proper nibble for generation of RXD[3:0] to the MII. The descrambler shall employ the polynomial defined in 147.3.2.5. An implementation of self-synchronizing descrambler by linear-feedback shift register is shown in figure TBD#2. The bits stored in the shift register delay line at time  $n$  are denoted by  $Dcr_n[16:0]$ . At every MII clock cycle, each bit of  $Sr_n[3:0]$  is shifted in as new  $Scr_n[0]$  and the descrambler is advanced by one bit. The output bit  $RXD[i]$  represented by the exclusive OR of  $Dcr_n[13]$ ,  $Dcr_n[16]$  and  $Dr_n[i]$  is generated, with  $i$  ranging from 0 to 3 (i.e. LSB first). The descrambler is reset upon execution of the PCS Reset function. If PCS Reset is executed, all bits of the 17-bit vector representing the self-synchronizing descrambler state are arbitrarily set. The initialization of the descrambler state is left to the implementer.





# #324 → Scrambler - Proposed text changes

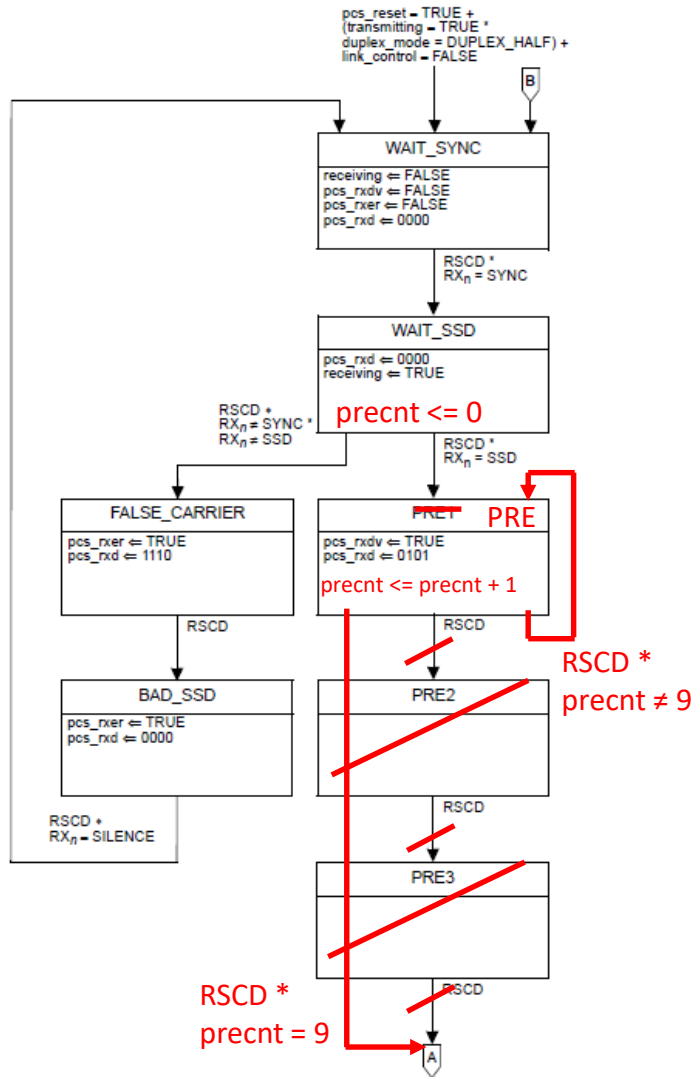


Figure 147-6—PCS Receive state diagram (1 of 2)

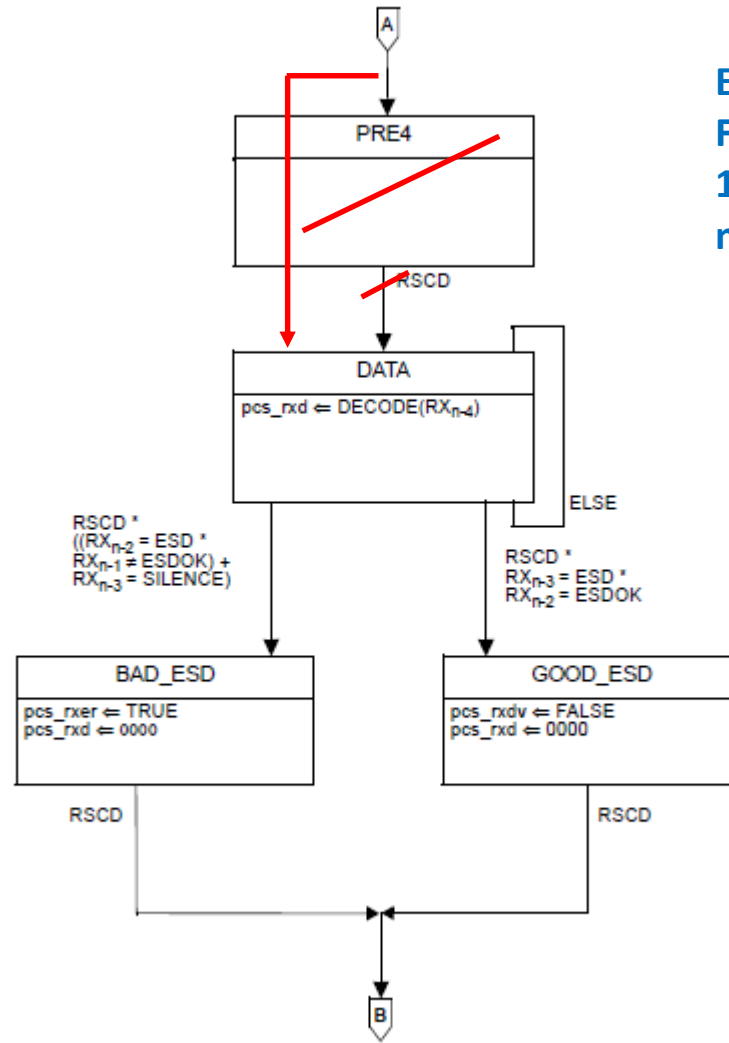


Figure 147-7—PCS Receive state diagram (2 of 2)

EDIT. NOTE:  
Fig 147-6 and  
147-7 could be  
merged

## #325 → subclause 148.4.6.1

During the COLLIDE state, the PLCA Data state machine asserts `packetPending = FALSE` and `CARRIER_STATUS = CARRIER_ON` via the `PLS_CARRIER.indication` primitive. When the MAC has finished to send the jam bits as described in Clause 4 it waits for the next transmit opportunity by switching to PENDING state.

During the PENDING state, the PLCA Data state machine asserts `packetPending = TRUE` and keeps `CARRIER_STATUS = CARRIER_ON` via the `PLS_CARRIER.indication` primitive to prevent the MAC to make new transmit attempts until PLCA Control state machine signals that a new transmit opportunity is met. At that point `CARRIER_STATUS` is set to `CARRIER_OFF` to have the MAC actually resend data after waiting one IPG period as described in Clause 4.

**Rationale:** In mis-configured networks physical collisions might happen. In such case setting packetPending flag in PLCA Data state machine in COLLIDE state may cause trouble (e.g. COMMITTING while JAMMING).

# #326 → Figure 148-6

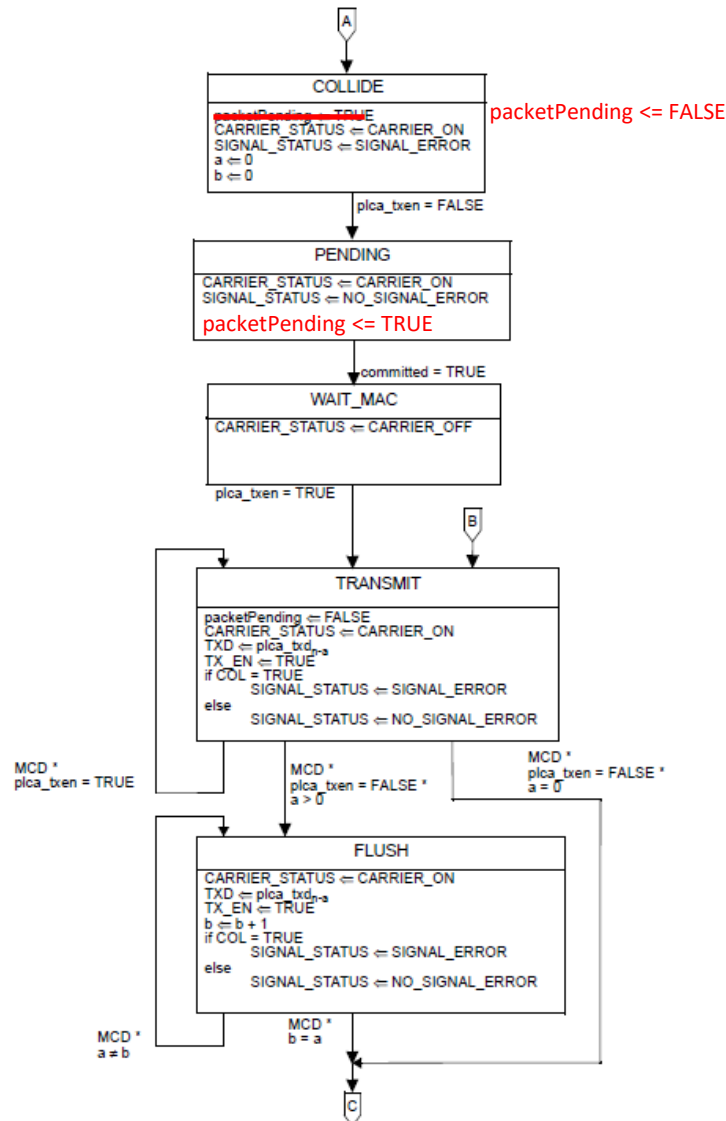


Figure 148-6—PLCA DATA state diagram (2 of 2)

**Rationale:** In mis-configured networks physical collisions might happen. In such case setting packetPending flag in PLCA Data state machine in COLLIDE state may cause trouble (e.g. COMMITTING while JAMMING).

## #327 → subclause 148.4.5.1

---

When PLCA functions are enabled, the PHY with `local_nodeID` variable set to 0 immediately switches to **RECOVER state and waits for all other PHYs to be silent for at least `RECV_BEACON_TIMER`**. Then it switches to **SEND\_BEACON** state to have all other PHYs synchronize their own transmit opportunity counter and related timer. Slave PHYs wait in **RESYNC** state until a BEACON is received.

**Rationale:** The node with ID = 0 could be reset in the middle of a BEACON cycle and start over sending a new BEACON while other PHYs are still in the process of transmitting / waiting their TO.

To avoid this the node with ID = 0 could start in recovery mode and wait for the media to be silent before sending the BEACON

# #328 → Figure 148-3

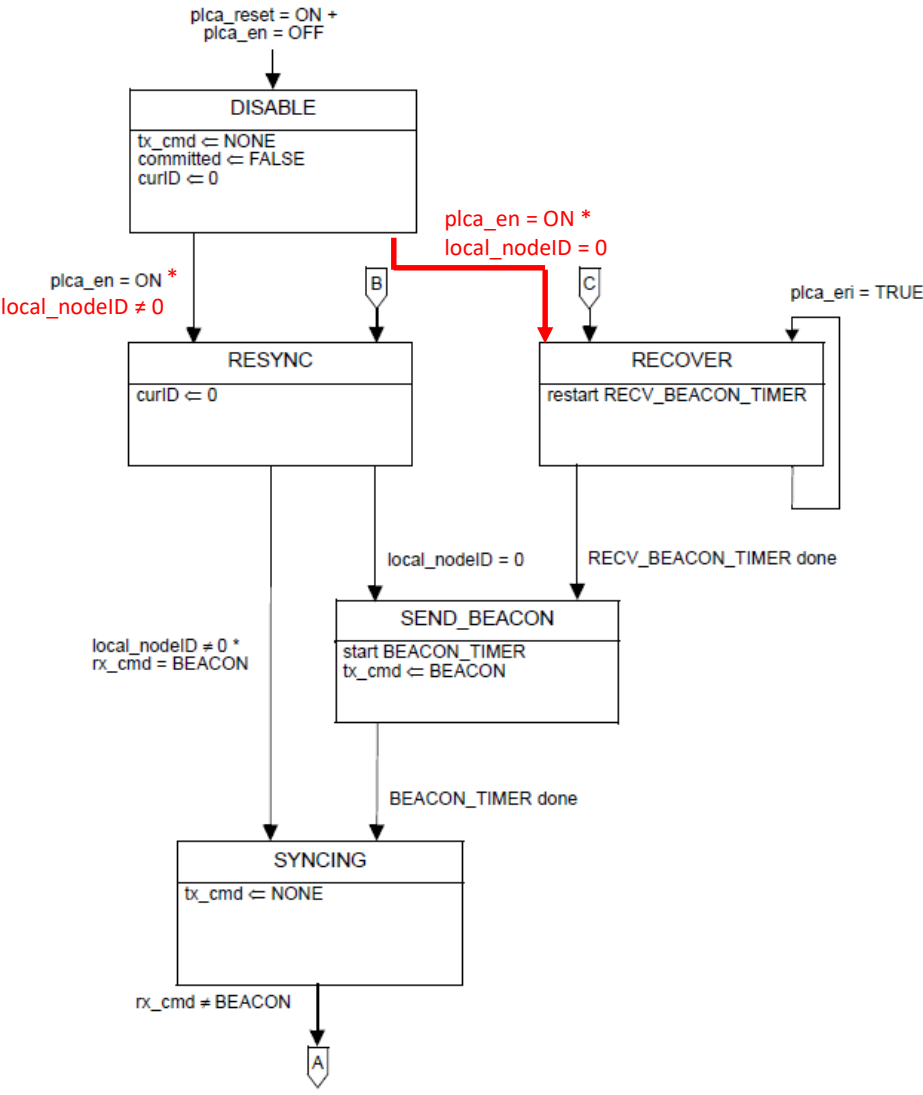


Figure 148-3—PLCA Control state diagram (1 of 2)

**Rationale:** The node with ID = 0 could be reset in the middle of a BEACON cycle and start over sending a new BEACON while other PHYs are still in the process of transmitting / waiting their TO. To avoid this the node with ID = 0 could start in recovery mode and wait for the media to be silent before sending the BEACON

## # 329 → Subclause 148.4.6.1

---

During the HOLD state the PLCA Control state machine is notified via the packetPending variable that data is available to be transmitted. At next transmit opportunity the PLCA Control state machine eventually allow transmitting the delayed data by setting the "committed" variable to TRUE. In such case the PLCA Data state machine switches to TRANSMIT state to actually deliver the data for the PHY to encode and transmit on the medium.

If TX\_ER is asserted during the HOLD state, the PLCA Data state machine switches to ABORT state to assert packetPending = FALSE and waits for the MAC to stop sending data. The aborted packet will not be transmitted on the medium.

If another PHY starts a transmission after meeting its own transmit opportunity, delayed data cannot be held anymore and a logical collision is triggered by switching to COLLIDE state.

## #330 (TX\_ER) → Subclause 148.4.6.2

---

**TX\_EN**                      The MII signal TX\_EN.

**TX\_ER**                      The MII signal TX\_ER.

**COL**                        The MII signal COL.

# #331 (TX\_ER) → Figure 148-5

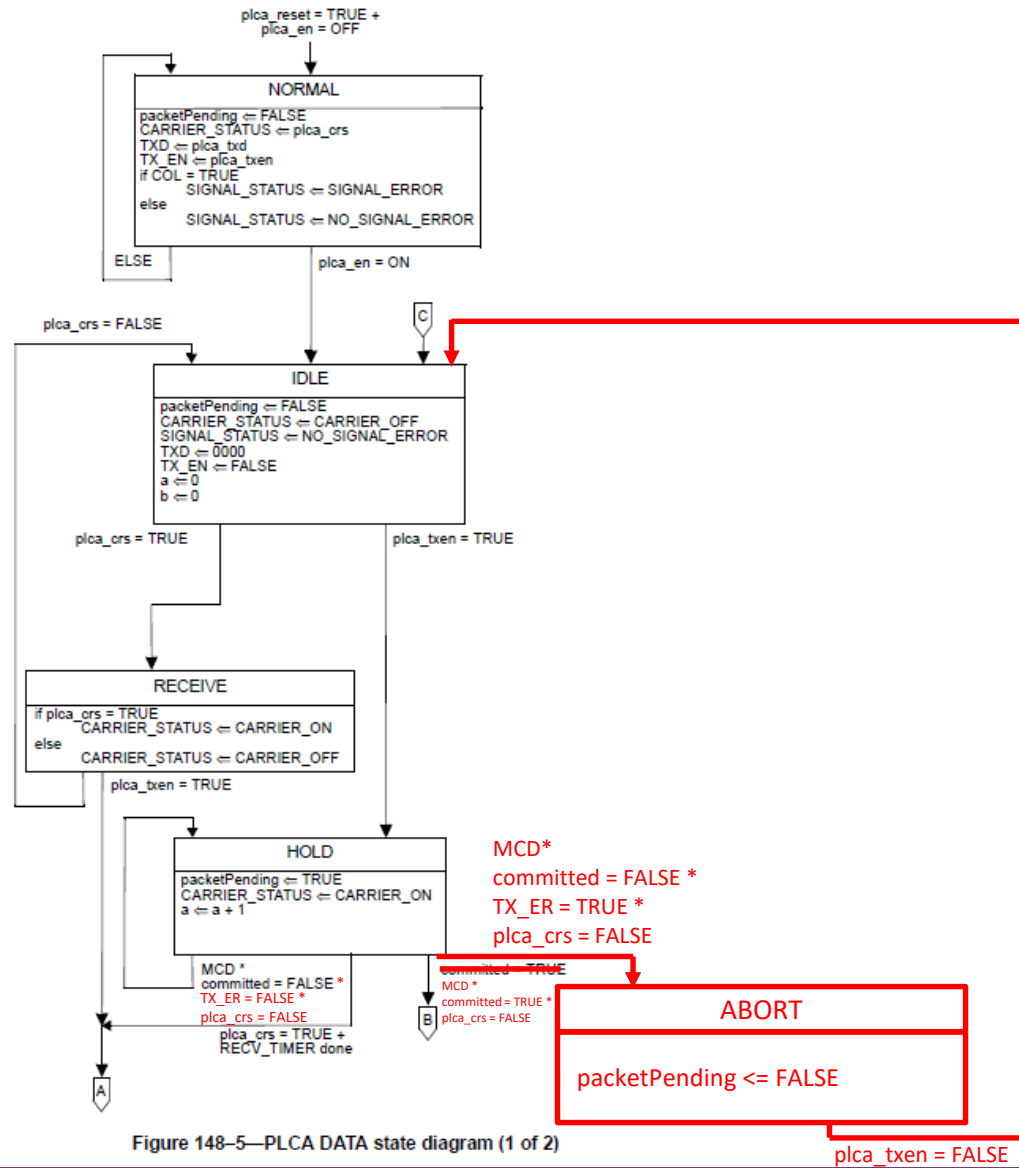


Figure 148-5—PLCA DATA state diagram (1 of 2)

plca\_txen = FALSE



The working principle of PLCA is that each PHY on a multidrop network is granted, in turn, a single transmit opportunity based on its assigned unique node ID.

At any time, only the PHY owning a transmit opportunity is allowed to send data over the medium, therefore avoiding physical collisions.

Transmit opportunities are generated in a round-robin fashion every time the PHY with node ID = 0 signals a BEACON on the medium, indicating the start of a new cycle. This can only happen after each PHY has been given exactly one transmit opportunity, thus ensuring media access fairness.

PLCA relies on CSMA/CD functions to have the MAC delay a transmissions until a transmit opportunity is met.