Consensus proposal for training state diagram

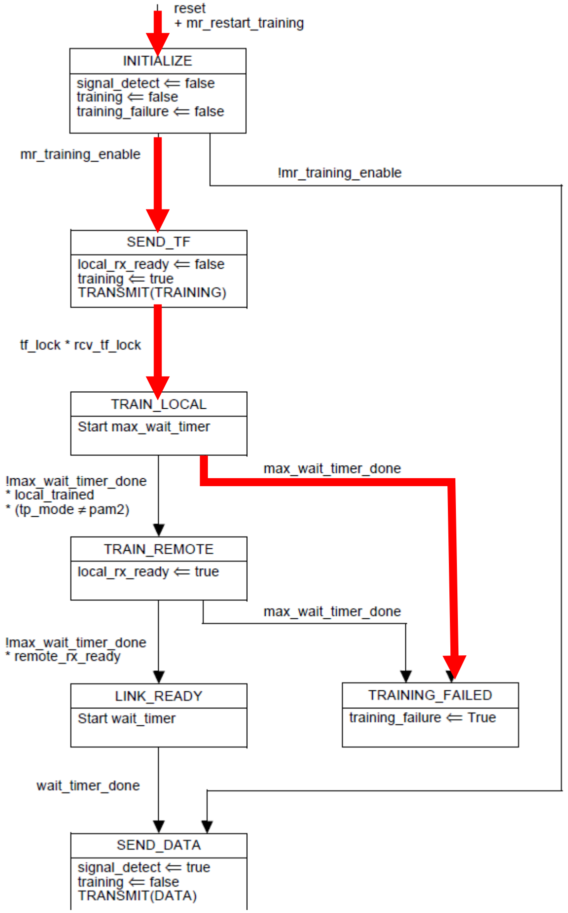
(supporting comments 118, 101, and 19)

Adee Ran, Intel

Oded Wertheim, Mellanox

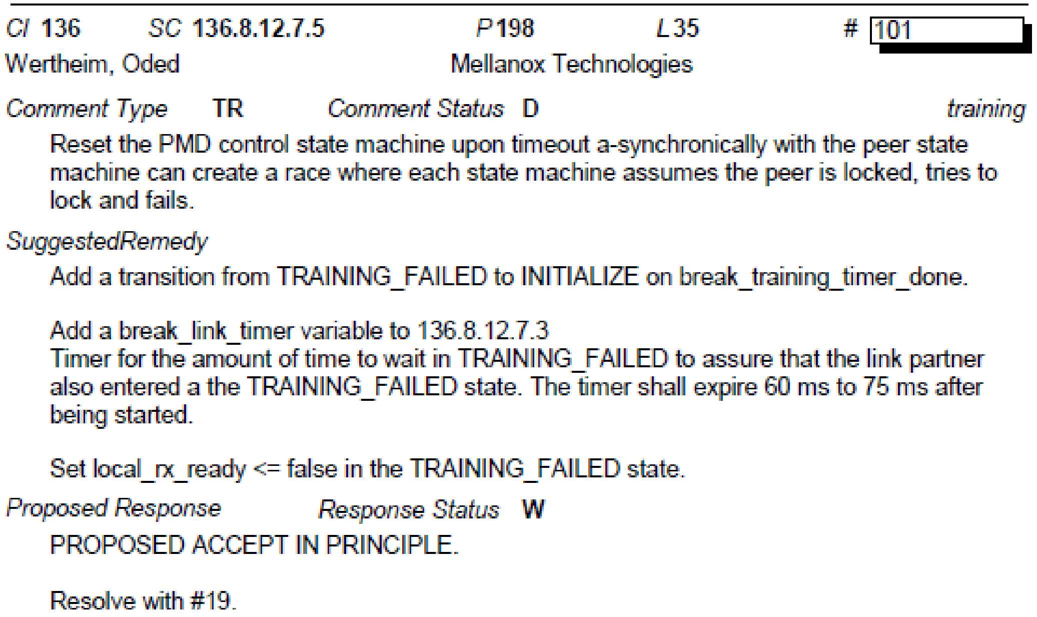
(with thanks to Jeff Slavick)

The state diagram – problem statement



- If one side gets to TRAINING_FAILED, it will cause the other side to also land there
  - With both sides transmitting TFs
- Existing this state is tricky…
  - Resetting only side A while side B is in TRAINING_FAILED would result in side A landing back in TRAINING_FAILED after max_wait_timer.
- You need to reset both sides within a short period
  - May not be feasible in a distributed environment…

Comment #101



Cl 136 *SC* 136.8.12.7.5 *P* 198 *L* 35 # 101

Wertheim, Oded Mellanox Technologies

Comment Type TR *Comment Status* D *training*

Reset the PMD control state machine upon timeout a-synchronically with the peer state machine can create a race where each state machine assumes the peer is locked, tries to lock and fails.

SuggestedRemedy

Add a transition from TRAINING_FAILED to INITIALIZE on break_training_timer_done.

Add a break_link_timer variable to 136.8.12.7.3
Timer for the amount of time to wait in TRAINING_FAILED to assure that the link partner also entered a the TRAINING_FAILED state. The timer shall expire 60 ms to 75 ms after being started.

Set local_rx_ready <= false in the TRAINING_FAILED state.

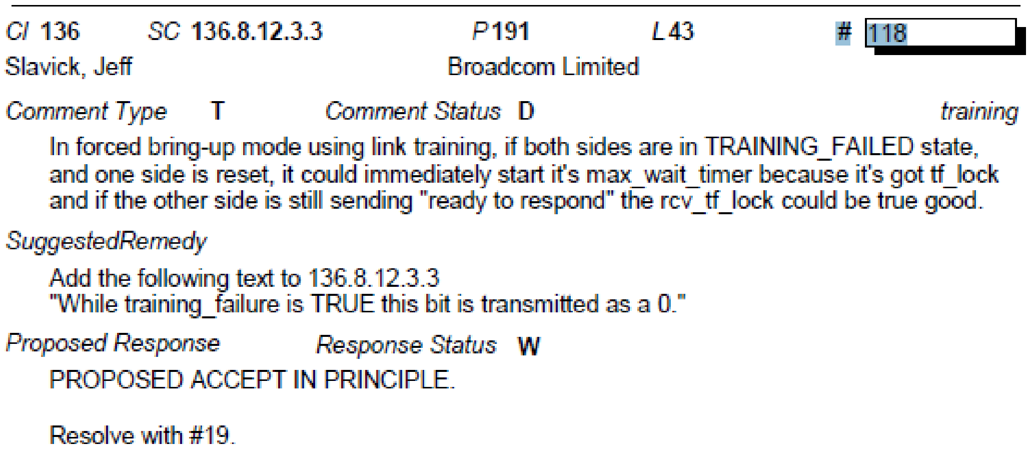
Proposed Response *Response Status* W

PROPOSED ACCEPT IN PRINCIPLE.

Resolve with #19.

- Elements:
  - Change state diagram flow to auto-recover from TRAINING_FAILED
  - Add timer
- Fully eliminates deadlock
  - But removes information about the failed state
- Does not require management intervention

Comment #118



Cl 136 *SC* 136.8.12.3.3 *P* 191 *L* 43 # 118

Slavick, Jeff Broadcom Limited

Comment Type T *Comment Status* D *training*

In forced bring-up mode using link training, if both sides are in TRAINING_FAILED state, and one side is reset, it could immediately start it's max_wait_timer because it's got tf_lock and if the other side is still sending "ready to respond" the rcv_tf_lock could be true good.

SuggestedRemedy

Add the following text to 136.8.12.3.3
"While training_failure is TRUE this bit is transmitted as a 0."

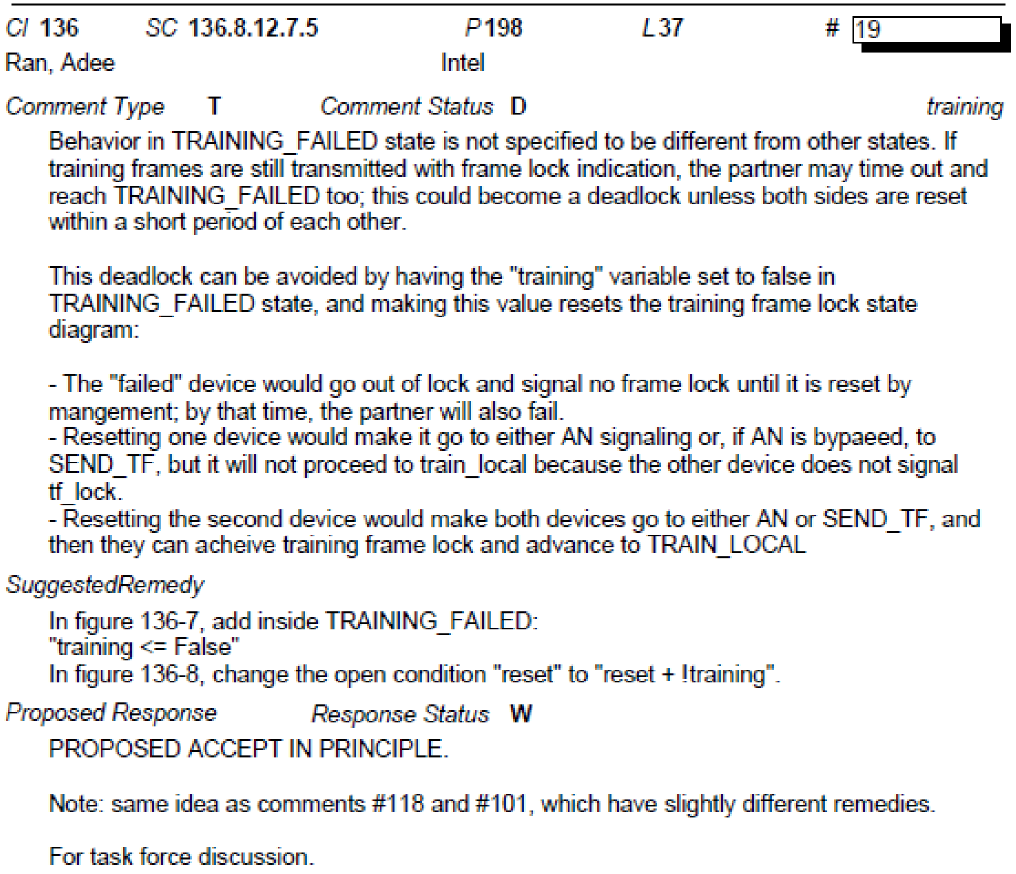
Proposed Response *Response Status* W

PROPOSED ACCEPT IN PRINCIPLE.

Resolve with #19.

- Mostly eliminates deadlock
  - But behavior is not obvious from state diagram
  - Deadlock may still occur if reset is too quick
- Requires management intervention
  - Will probably happen anyway

Comment #19



Cl 136 *SC* 136.8.12.7.5 *P* 198 *L* 37 # 19

Ran, Adee Intel

Comment Type T *Comment Status* D *training*

Behavior in TRAINING_FAILED state is not specified to be different from other states. If training frames are still transmitted with frame lock indication, the partner may time out and reach TRAINING_FAILED too; this could become a deadlock unless both sides are reset within a short period of each other.

This deadlock can be avoided by having the "training" variable set to false in TRAINING_FAILED state, and making this value resets the training frame lock state diagram:

- The "failed" device would go out of lock and signal no frame lock until it is reset by mangement; by that time, the partner will also fail.
- Resetting one device would make it go to either AN signaling or, if AN is bypaeed, to SEND_TF, but it will not proceed to train_local because the other device does not signal tf_lock.
- Resetting the second device would make both devices go to either AN or SEND_TF, and then they can acheive training frame lock and advance to TRAIN_LOCAL

SuggestedRemedy

In figure 136-7, add inside TRAINING_FAILED:
"training <= False"
In figure 136-8, change the open condition "reset" to "reset + !training".

Proposed Response *Response Status* W

PROPOSED ACCEPT IN PRINCIPLE.

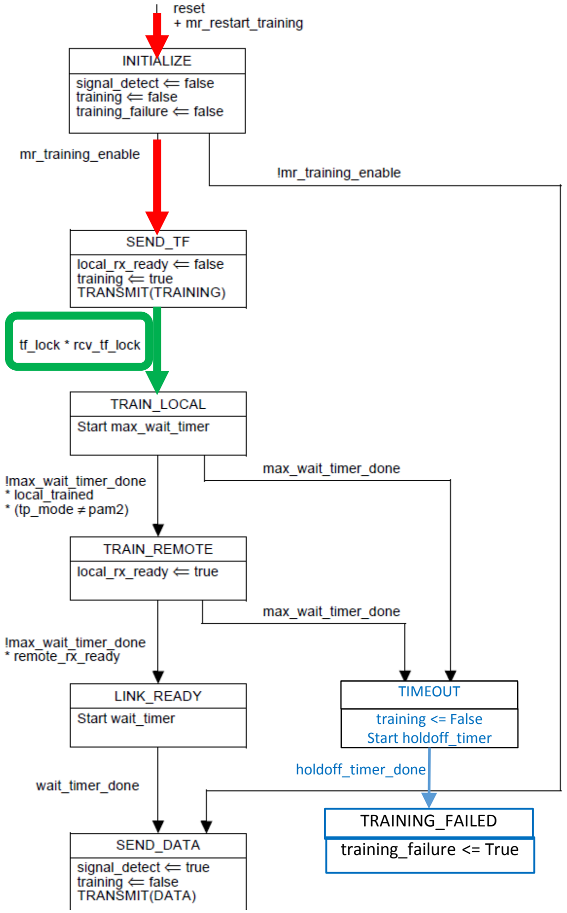
Note: same idea as comments #118 and #101, which have slightly different remedies.

For task force discussion.

- Mostly eliminates deadlock
  - Deadlock may still occur if reset is too quick
- Behavior fully specified in state diagrams
- Requires management intervention
  - Will probably happen
- Losing the frame alignment prevents the receiver from viewing the partner's state
  - This info could be useful for debugging

New proposal

- Based on suggested remedy to comment #19
  - Desire is to prevent losing the frame alignment in TRAINING_FAILED
- Instead of resetting the frame lock state diagram (Figure 136-8), we can just stop reporting to the partner that we are locked, by changing the definition of tf_lock to include the “training” state
  - This change will keep the frame alignment, but the partner will see “remote_tf_lock=false” until TRAINING_FAILED is exited (by reset)
  - After one partner is reset, it may re-lock, but it will stop in SEND_TF while partner is still in TRAINING_FRAME
  - The two partners need to “meet” by both being in SEND_TF, and reporting lock state to each other. Then they start their timers together
- To prevent deadlock if reset is asserted too quickly, add a timer to hold off asserting training_failure
  - Timer should be longer than the variation of max_wait_timer, which is ±30 ms
- Proposal: update the PMD control state diagram based on the diagram on the right. Change clause text as shown in the next slide.



Suggested remedy – text changes

- In 136.8.12.7.1, change the definition of tf_lock
  - FROM “Boolean variable that is true when the training frame marker positions have been identified and is false otherwise”
  - TO “Boolean variable that is true if the value of training is true and training frame marker positions have been identified, and is false otherwise”
- In 136.8.12.7.3, add new timer
  - holdoff_timer: This timer is started when the PMD control state diagram enters the TIMEOUT state. The terminal count of this holdoff_timer is 40 ms ± 2%.