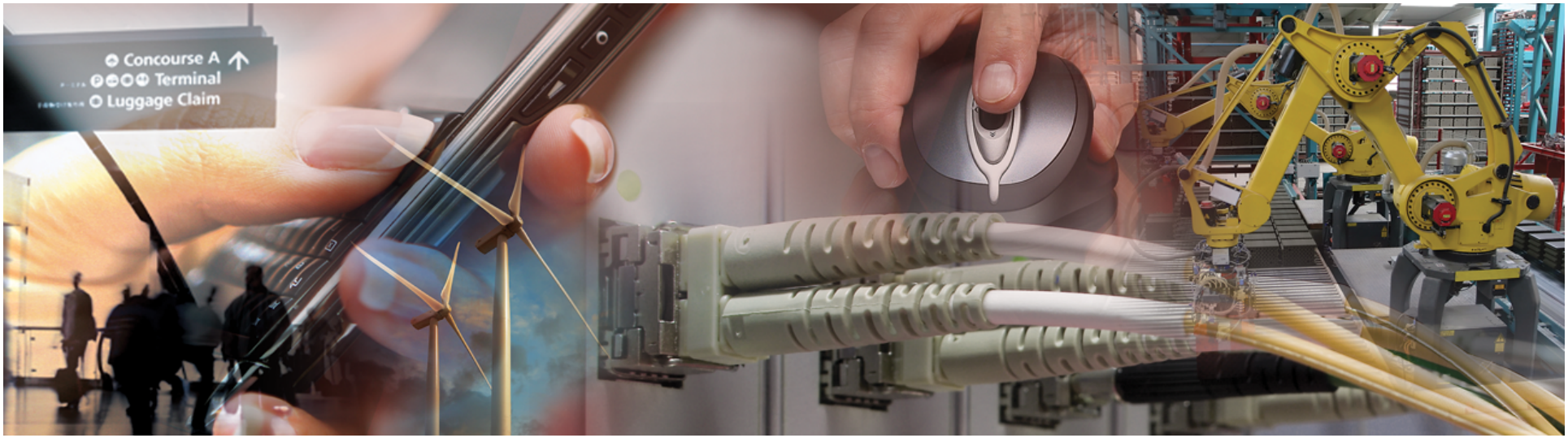
Supporting materials for comment #123



Jeff Slavick

AVAGO TECHNOLOGIES

Comment #123

Cl **108** *SC* **180.5.4.5** *P* **116** *L* **15** # 123

Slavick, Jeff — Avago Technologies

Comment Type **TR** *Comment Status* **X**

With a Clause 49 LPI state diagram you can exit LPI state without ever going to sleep (path from SLEEP -> ACTIVE exists).

lpi_rapid_align is set to true whenever rx_lpi_active is set to true, which occurs when /LI/ are seen. So if the Tx sends some /LI/ but doesn't actually go to sleep lpi_rapid_align could be set. The only way to clear lpi_rapid_align is to successfully achieve alignment with rapid CWMs.

Addiitionally while in the 2_GOOD state you would reset the lpi_rapid_align setting baed on rx_lpi_active being TRUE, if it changes to FALSE (transition from /I/ to /LI/ during WAKE) then you'd also end up stuck trying to frame to rapid CWMs

SuggestedRemedy

Add the assignment of lpi_rapid_align <= rx_lpi_active in the 2_GOOD state of Figure 108-6

Change the definition for lpi_rapid_align to be:
Boolean variable that is set according to the FEC synchronization state diagram in Figure 108 6.

Add a WAKE_FAIL state to Figure 108-6 which is entered if the hold-off timer defined in 108.5.3.7 expires and sets lpi_rapid_align <= false and transitions to the LOCK_INIT state via a UCT transition

Create a definition for the Rx EEE hold_timer in 108.5.4 to be referenced by Figure 108-6 and 108.5.3.7

Proposed Response *Response Status* **O**

D2.0 definitions

lpi_rapid_align

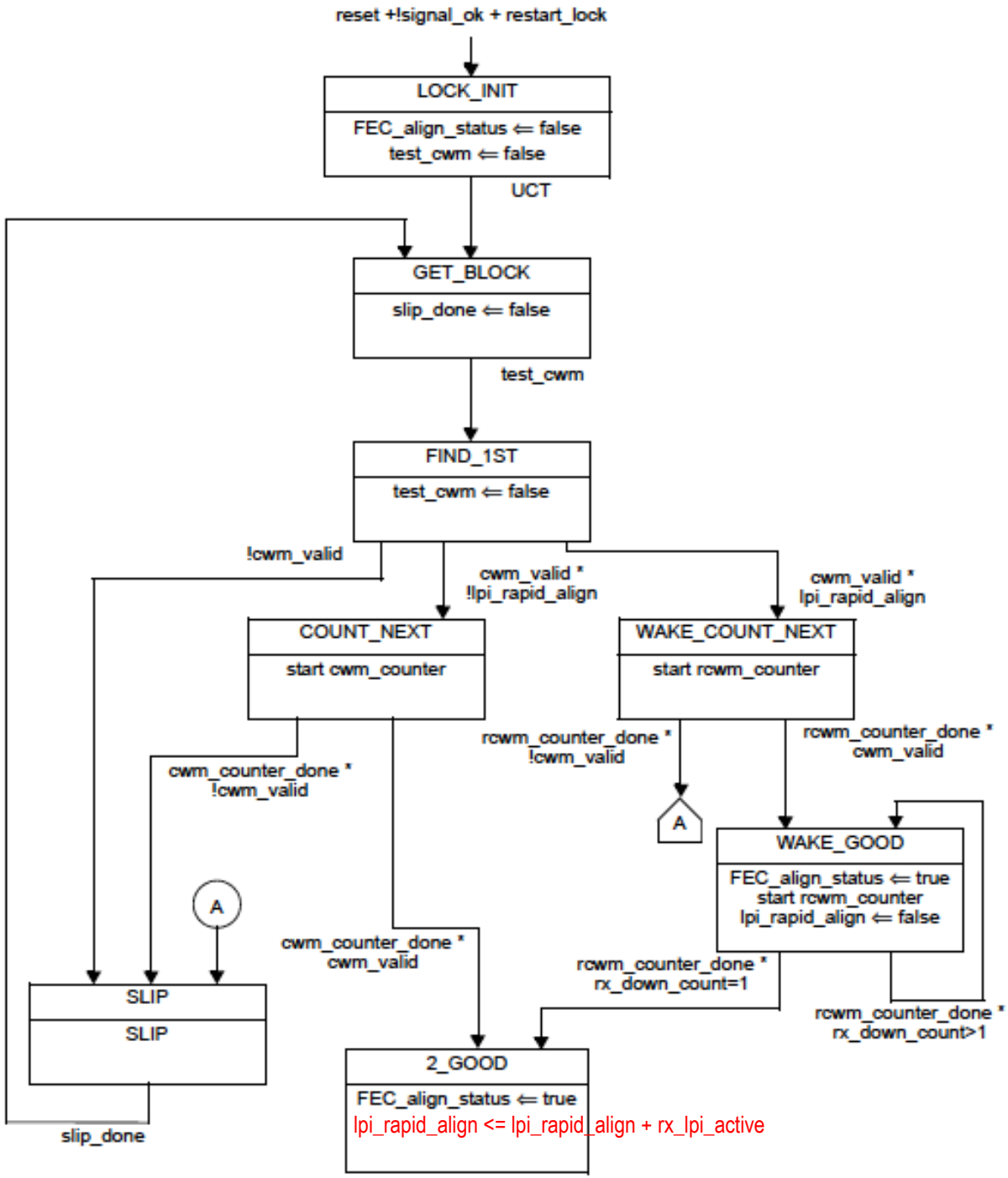
Boolean variable that is set to true when rx_lpi_active is true, and is set to false according to the FEC synchronization state diagram in Figure 108–6.

rx_lpi_active

Boolean variable that is set to true when the rate compensation for codeword markers in the receive direction function (108.5.3.6) infers that the Low Power Idle is being received from the link partner, and is set to false otherwise.

- This means lpi_rapid_align is a latched-high version of rx_lpi_active that only clears when you enter WAKE_GOOD state of Figure 108-6

D2.0 Figure 108-6



Clause 49

- In the Clause 49 PCS you can begin sending /LI/ and then go back to ACTIVE without ever going to sleep.
- This means 108 could receive some /LI/ which will set rx_lpi_active high, latch the lpi_rapid_align variable TRUE, then see /I/ come through and de-assert rx_lpi_active, but lpi_rapid_align stays TRUE and the link doesn't go to sleep.

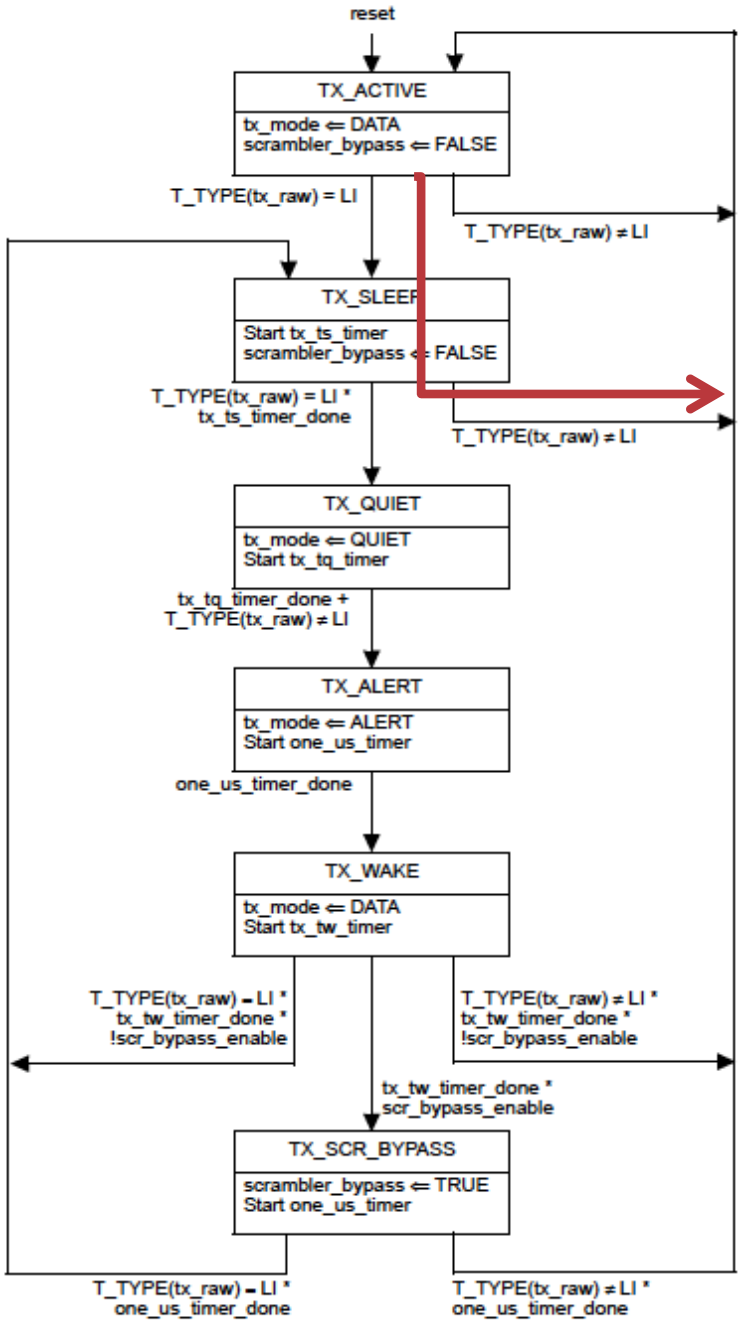


Figure 49–12—LPI Transmit state diagram

Clause 49

- Also during the WAKE process you could start with /LI/ being sent in the TX_WAKE state and then change it to /I/ in the TX_WAKE or TX_SLEEP state and return to TX_ACTIVE.
- This also causes lpi_rapid_align to be latched TRUE when the Tx transitions from /LI/ to /I/ after the Rx has reached 2_GOOD state

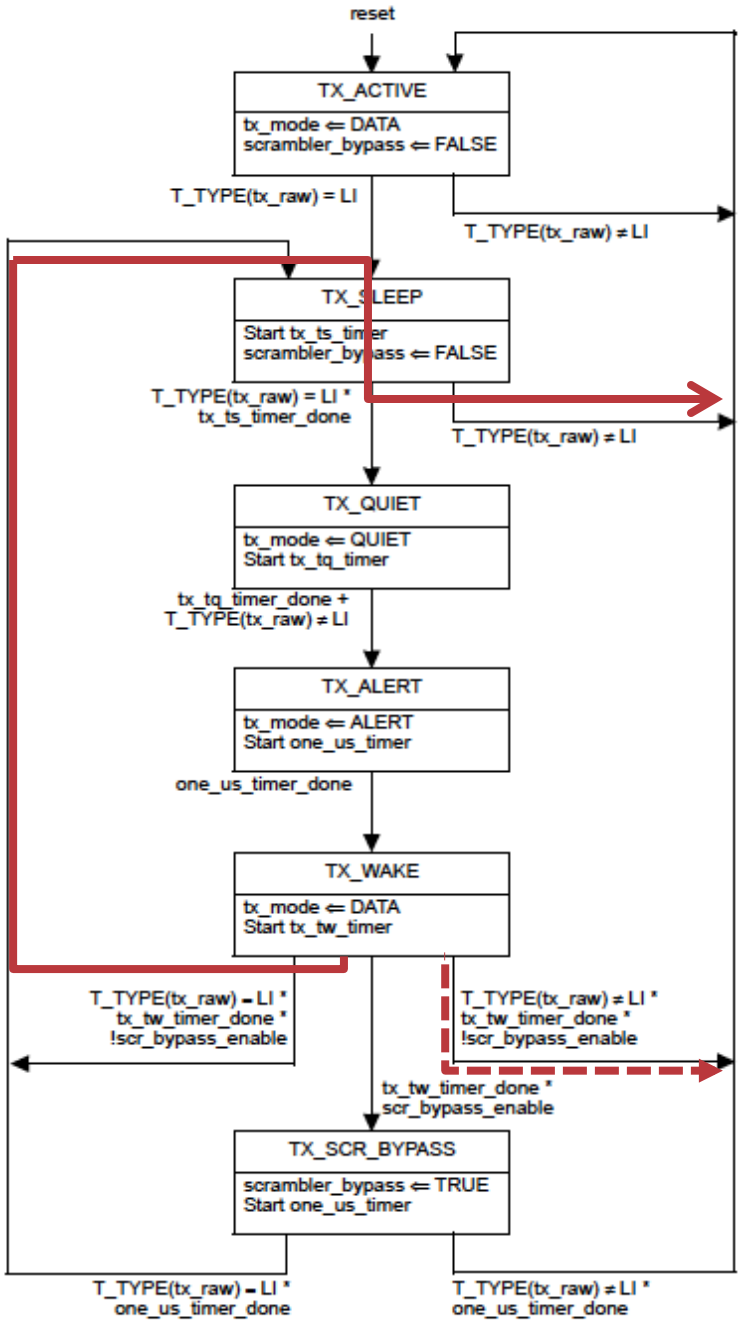


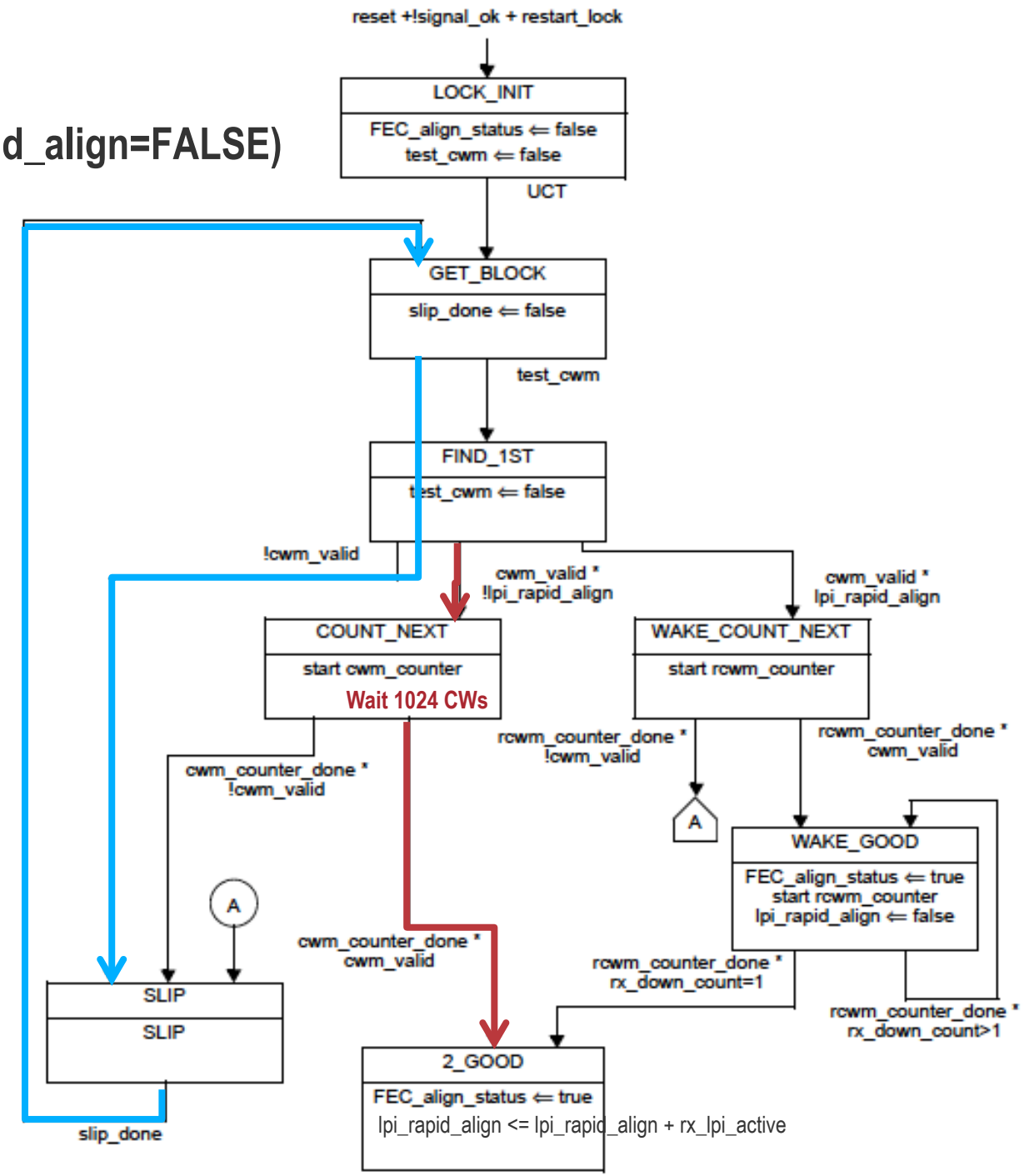
Figure 49–12—LPI Transmit state diagram

Issue

- **lpi_rapid_align controls how often CWMs are checked for**
  - TRUE = every RS-FEC codeword, FALSE = every 1024th
- **Once lpi_rapid_align == TRUE the only way to ever set it FALSE is to lock onto RCWM.**
- **If frame_lock is lost with lpi_rapid_align == TRUE and RCWMs aren't sent, then there's no method to clear lpi_rapid_align and try to lock at the normal CWM interval**
  - Ie. Lock could be lost due to
    - Failure to lock to RCWMs before they disappear
    - Excessive uncorrectable codwords
    - Remote tx turns off then on for a non-SLEEP reason (reset, cable pull)

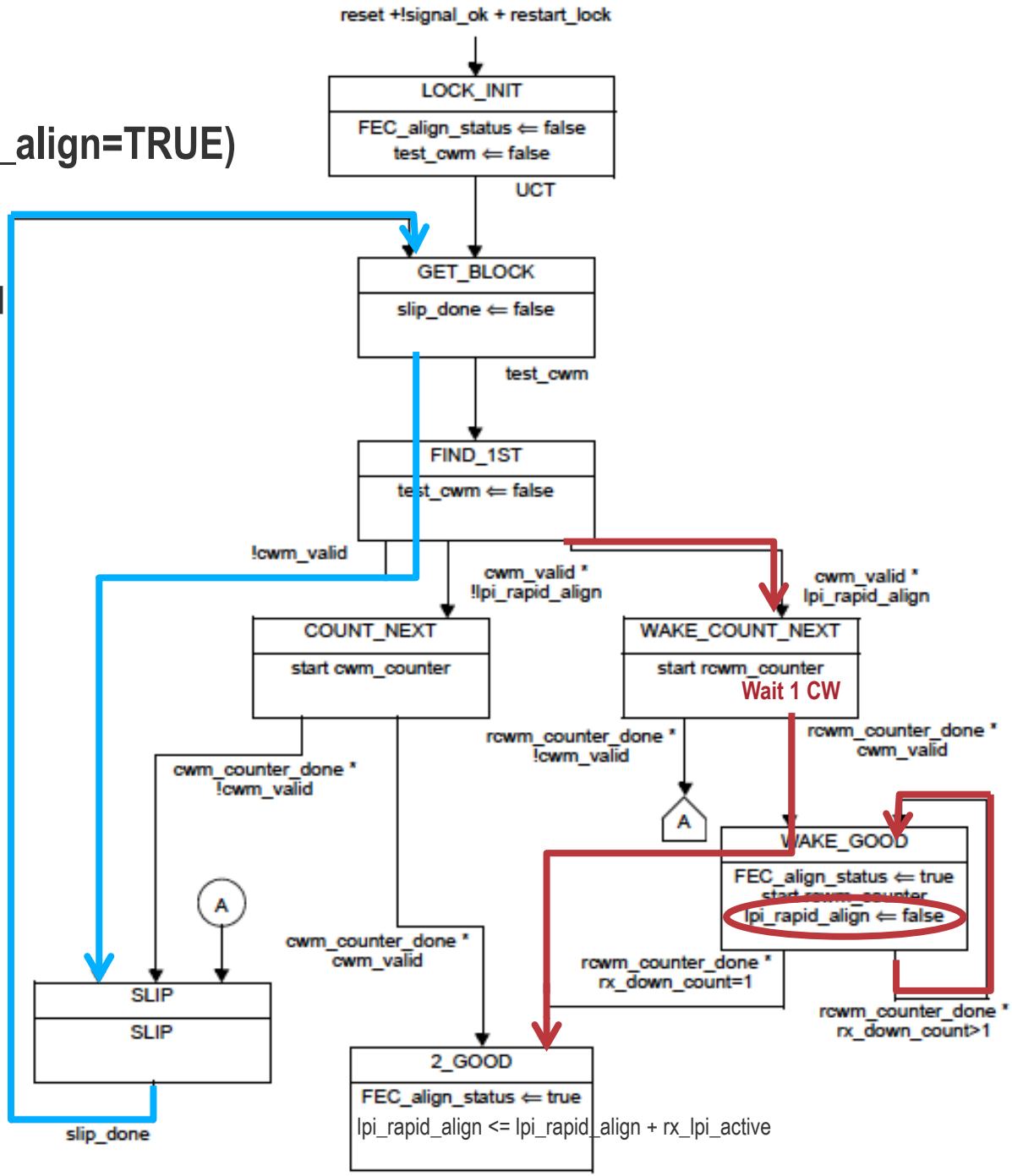
Flow Taken (lpi_rapid_align=FALSE)

- Blue is taken until you locate the 1st CWM, then RED is taken to confirm lock



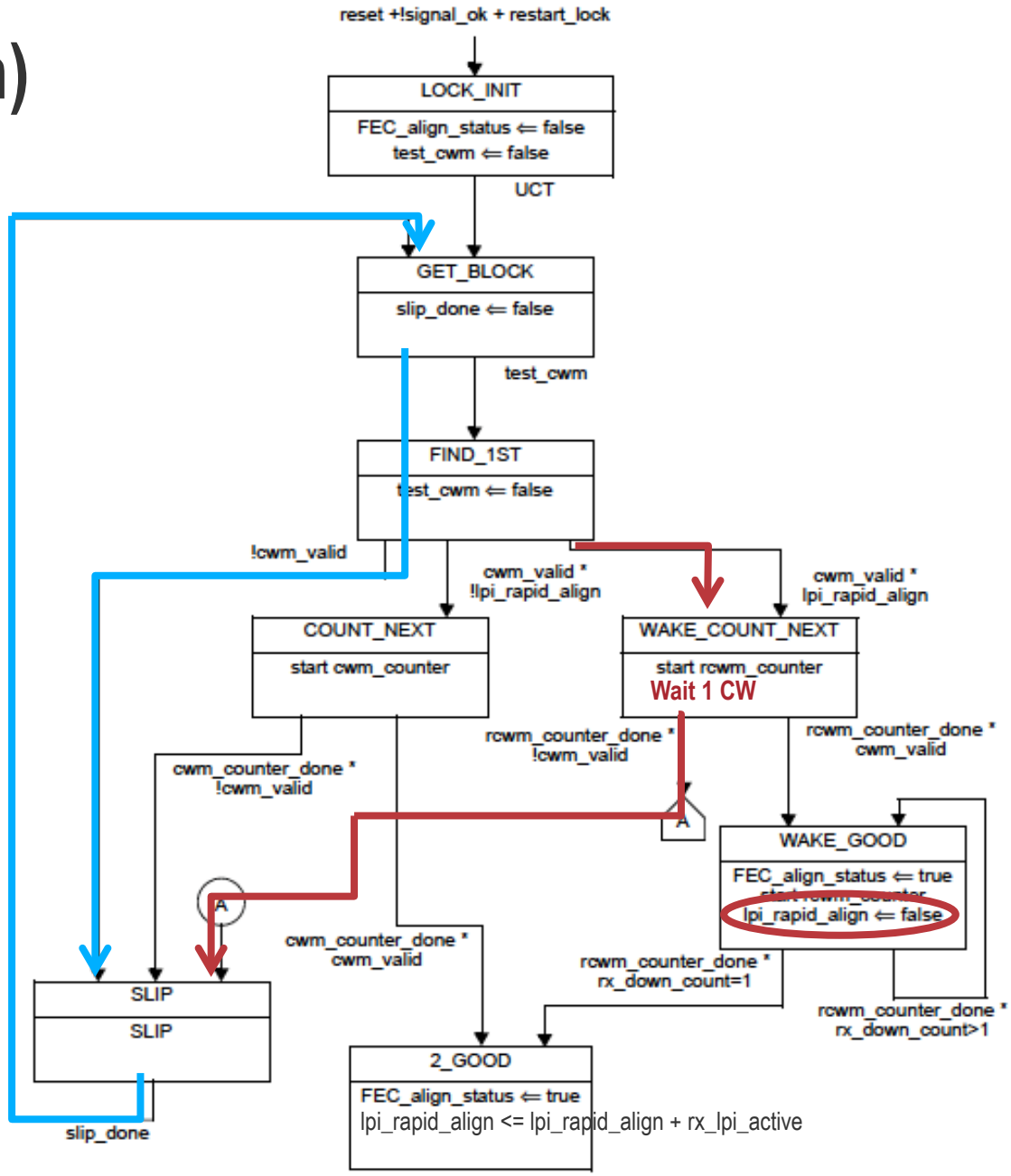
Flow Taken (lpi_rapid_align=TRUE)

- Blue is taken until you locate the 1st RCWM, then RED is taken to confirm lock
- lpi_rapid_disable is cleared only after confirming lock
- Note that lpi_rapid_disable could be set immediately back to TRUE if /LI/ is being received



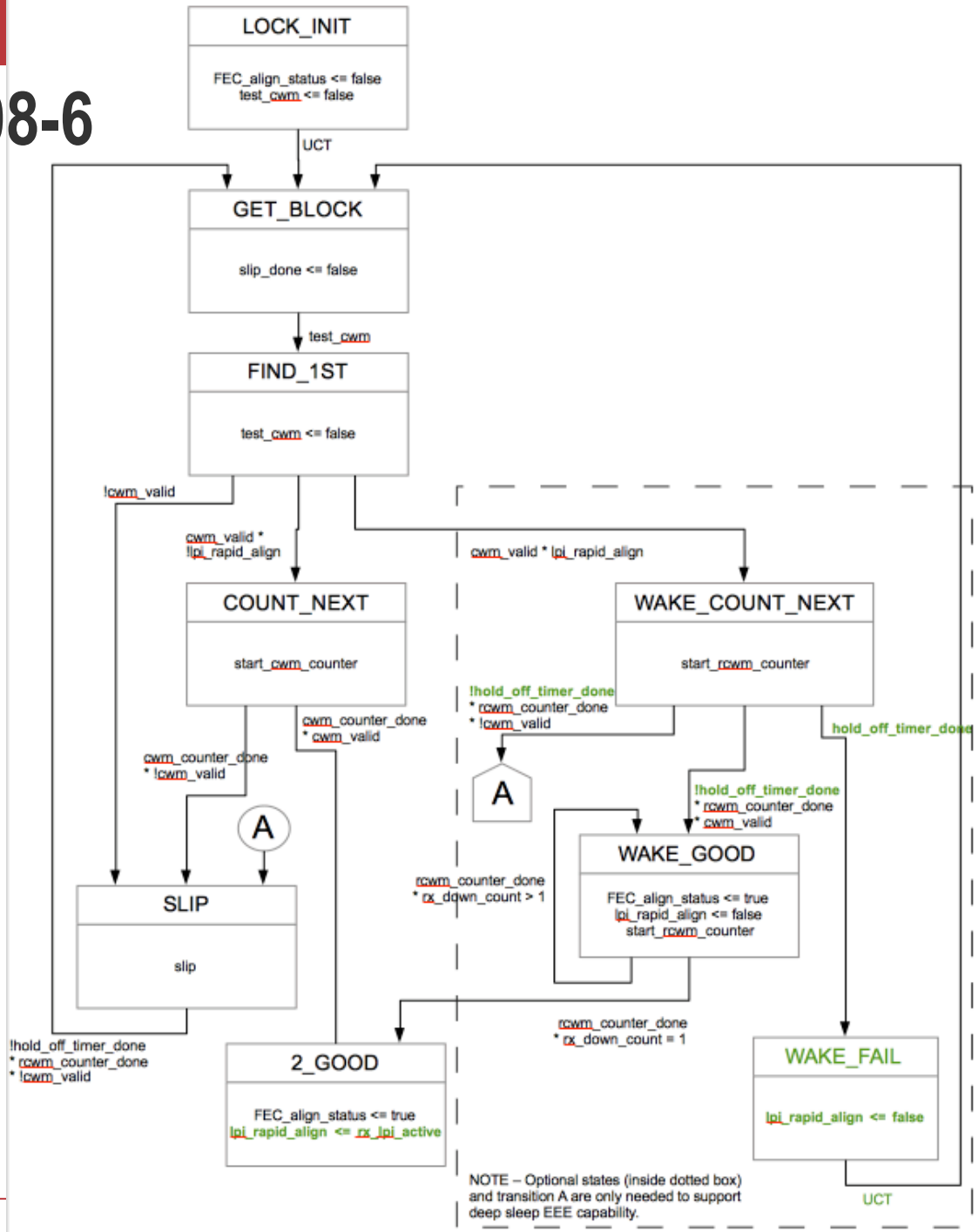
Flow Taken (broken)

- lpi_rapid_align == TRUE, TX is in TX_ACTIVE state, Rx loses lock
- Blue is taken until you locate the 1st CWM/RCWM, then RED is taken to confirm lock
- Tx is sending CWM at 1024 interval instead of every CW, SM is stuck in infinite loop.
- Tx and Rx aren't in the same EEE state and there's no timeout in Rx to revert to non-EEE mode



Remedy to Figure 108-6

- **Add WAKE_FAIL state to reset lpi_rapid_align if the hold_off_timer expires.**
  - Resets RX EEE mode if Tx and Rx get out of sync
- **Update lpi_rapid_align based on current state of rx_lpi_active while in 2_GOOD state**
  - Rx tracks changes in /I/ and /LI/ from Tx to reduce probability WAKE_FAIL is ever visited



Remedy – Definition updates

- **lpi_rapid_align**
  - Boolean variable set according to the FEC Synchronization state machine in Figure 108-6. Set to false when reset is TRUE.
- **108.5.4.3a Timers**
- **The following timer is only used for EEE deep sleep capability.**
- **hold_off_timer**
  - This timer starts when rx_mode (or rx_tx_mode if appropriate) transitions from QUIET to DATA. The timer is stopped when the terminal count is reached or the FEC Synchronization state machine enters the 2_GOOD state. The timer terminal count is set to 11.5μs. When the timer reaches terminal count hold_off_timer_done is set to TRUE.

Remedy – other edits

- **Add a reference to the newly created timer sub-section into 108.5.3.7 item a)**
  - 108.5.3.7 is the RX Rapid codeword lock for EEE deep sleep description section. Item a) states to start a hold off timer.