ANALOG DEVICES
AHEAD OF WHAT'S POSSIBLE™

IEEE *Draft* P802.3cg/D3.0 10BASE-T1L LPI synchronization

Proposal relating to comment i-285

MICK MCCARTHY

22 MAY 2019

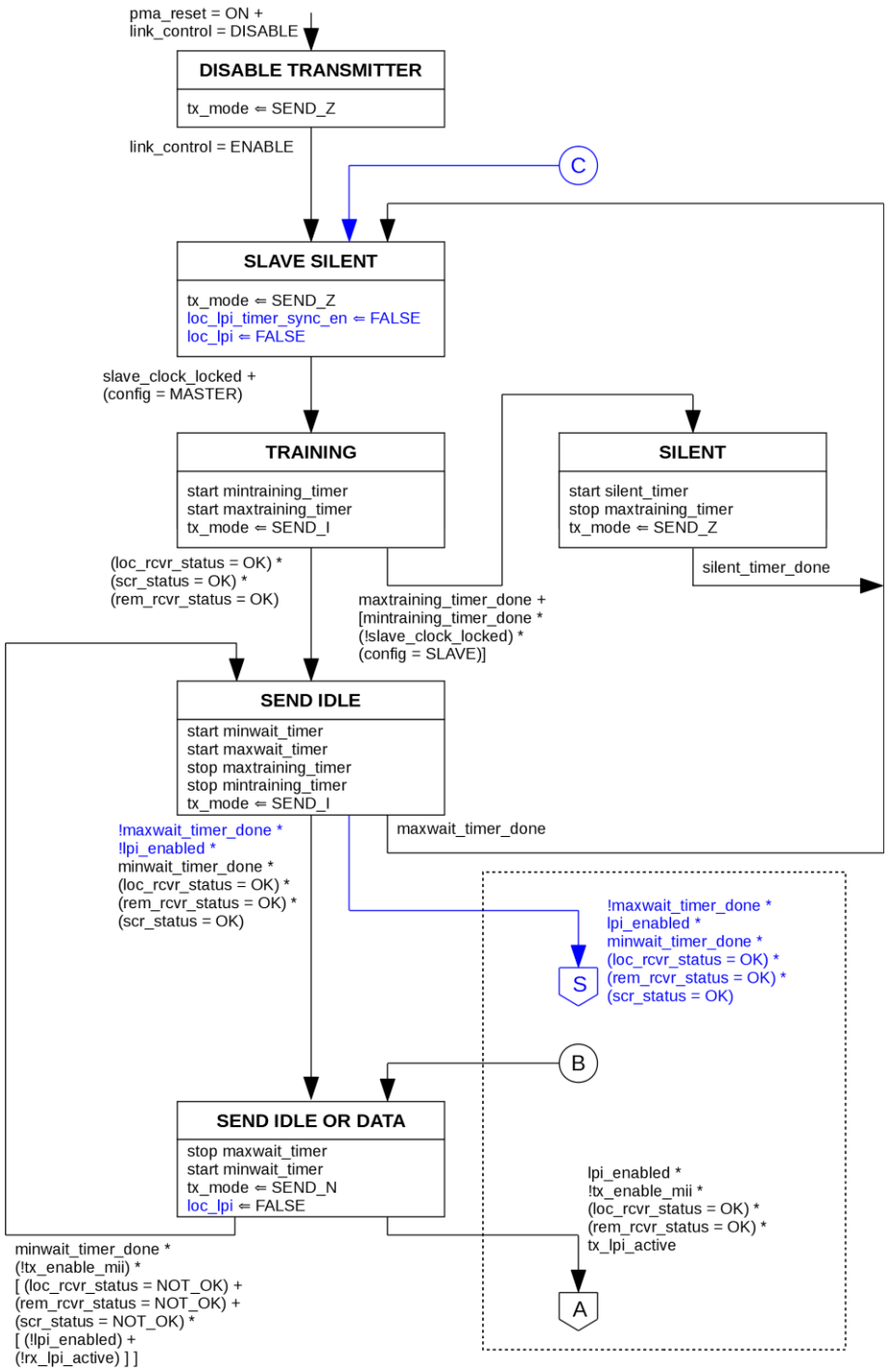
LPI QUIET REFRESH synchronization

- 10BASE-T1L LPI includes no mechanism to provide a PHY with clear timing information about when LPI QUIET and REFRESH modes will be entered and exited.
- This has potential to cause difficulty for 10BASE-T1L EEE PHY implementations. LPI modes will be entered and exited depending on data traffic.
- Other PHY technologies that employ of MASTER/SLAVE timing scheme and echo cancellation employ some kind of synchronization such that PHY implementations enjoy more certainty about when LPI modes are entered and exited.
  - 1000BASE-T symmetric LPI
  - 1000BASE-T1 asymmetric LPI, with LPI synchronization (subclause 97.3.5.1)

Overview of proposal

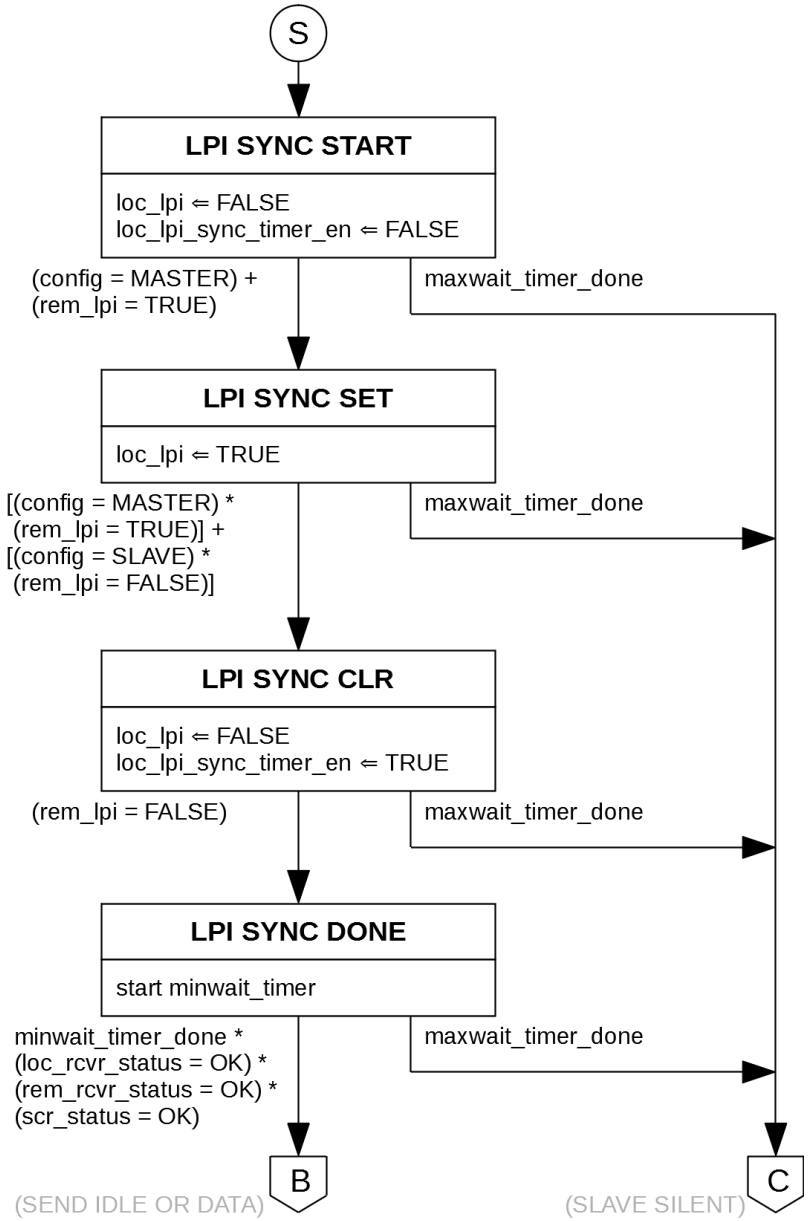
- Modify 10BASE-T1L PHY Control state diagram to add LPI synchronization mechanism using loc_lpi_req signal in advance of SEND IDLE OR DATA (link up).
- LPI synchronization mechanism dictates when a new LPI QUIET REFRESH timing state machine starts.
- Start of LPI QUIET REFRESH timing is communicated to link partner using loc_lpi_req signal, i.e. observed in link partner as rem_lpi_req
- LPI QUIET REFRESH timing state machine would remain active for the lifetime of the link.
- LPI QUIET REFRESH timing would synchronize to the symbol timer (TX_TCLK).
- A PHY implementation could take advantage of LPI synchronization:
  - To know when the link partner PHY is in the REFRESH state, and can restrict channel equalizer coefficient adaptation to only be active during this window.
  - To know when local PHY is in the REFRESH state, and can restrict echo canceller coefficient adaptation to only be active during this window.

Figure 146-14 – PHY Control state diagram (part a)



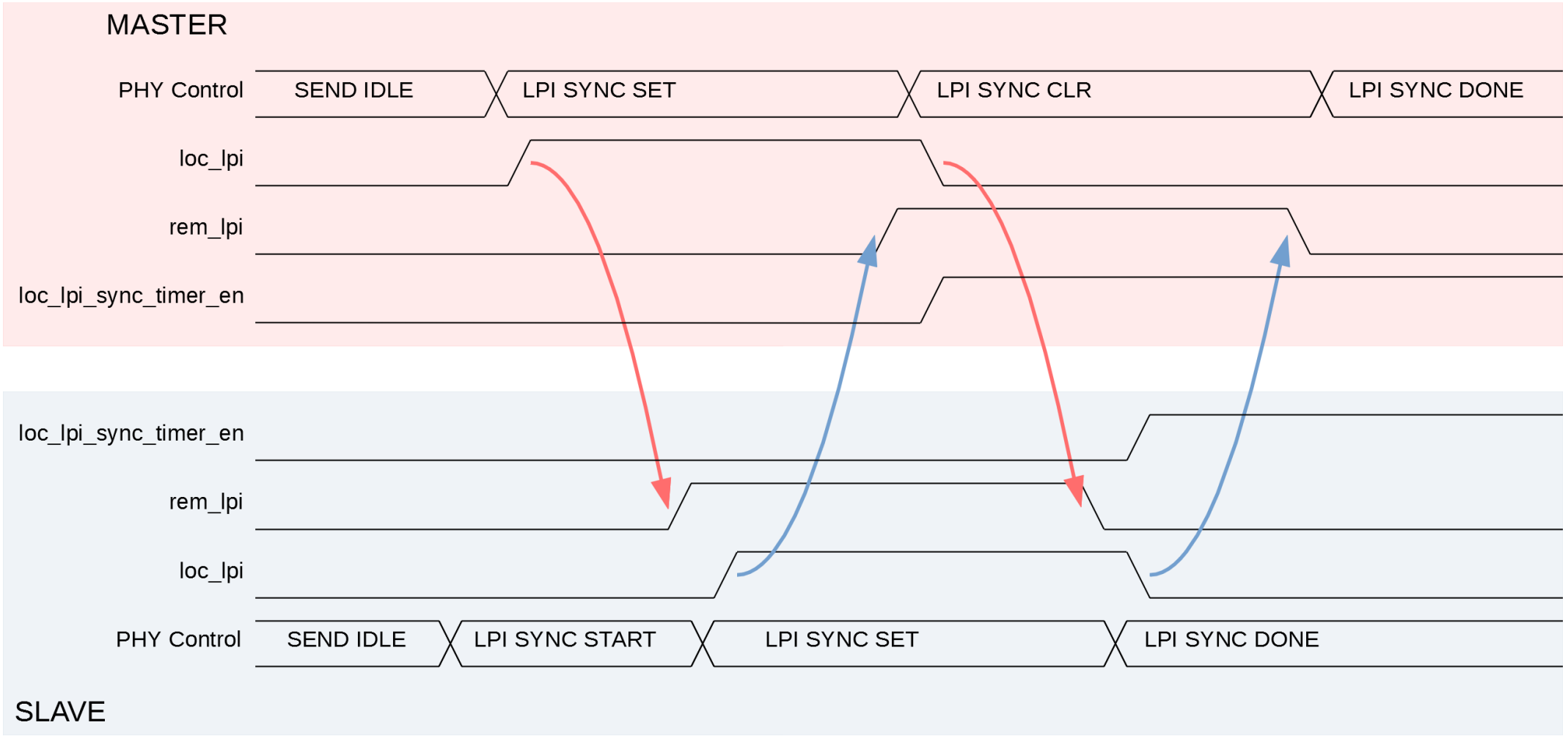
- Added entry to LPI synchronization sequence on exit from SEND IDLE state for LPI mode (labelled ‘S’ in diagram)
- New output from PHY Control state diagram: loc_lpi_timer_sync_en
  - TRUE enables local LPI synchronization timing
  - FALSE disables local LPI synchronization timing
  - Not encoded in transmit symbol stream; no new communicated parameters required
- Note some minor corrections
  - Use of !maxwait_timer_done in exit transitions from SEND IDLE
  - Renaming loc_lpi_req as loc_lpi (see later explanation)

PHY Control LPI synchronization

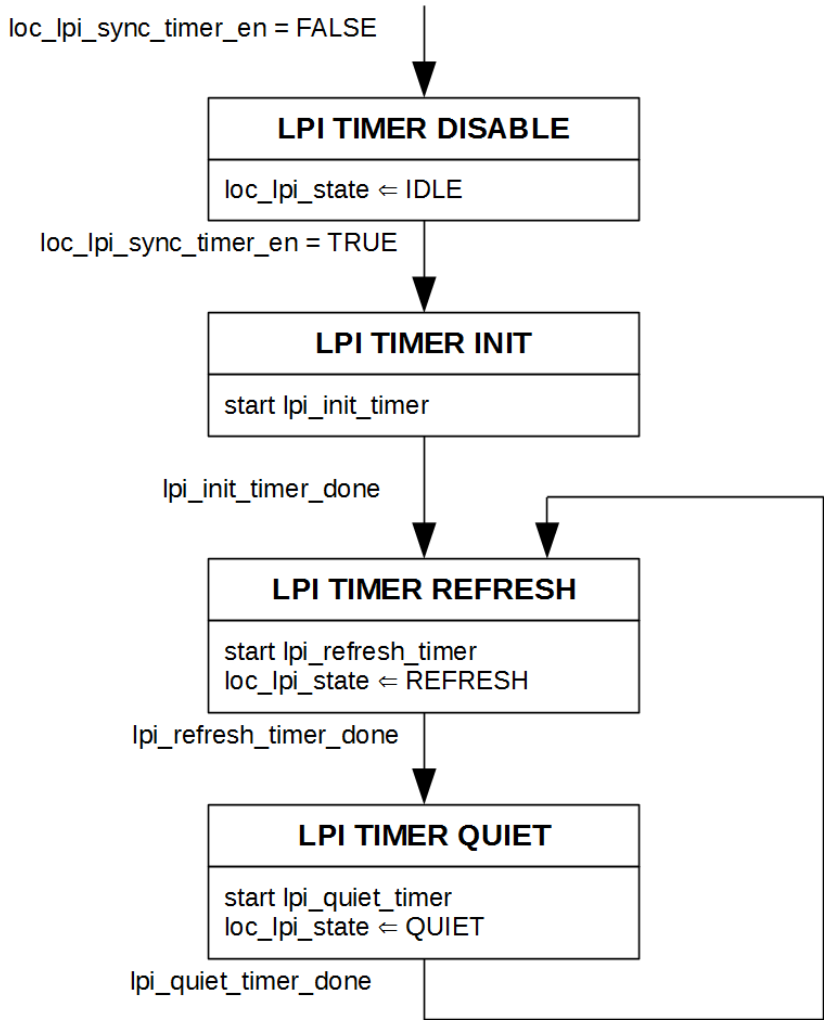


- LPI synchronization sequence is described as follows:
  - MASTER first sets loc_lpi = TRUE
  - SLAVE waits to see rem_lpi = TRUE (from MASTER), and then sets loc_lpi = TRUE
  - MASTER waits to see rem_lpi = TRUE (from SLAVE), and then sets loc_lpi = FALSE
  - MASTER also sets loc_lpi_sync_timer_en = TRUE at the same time
  - SLAVE waits to see rem_lpi = FALSE (from MASTER), and then sets loc_lpi = FALSE
  - SLAVE also sets loc_lpi_sync_timer_en = TRUE at the same time
- In both MASTER and SLAVE, transition of loc_lpi from TRUE to FALSE occurs at the same time as start of local LPI synchronization timing (loc_lpi_sync_timer_en transition from FALSE to TRUE).
- LPI synchronization mechanism takes place before link startup completion (PHY Control transition to SEND IDLE OR DATA).

LPI synchronization timing diagram

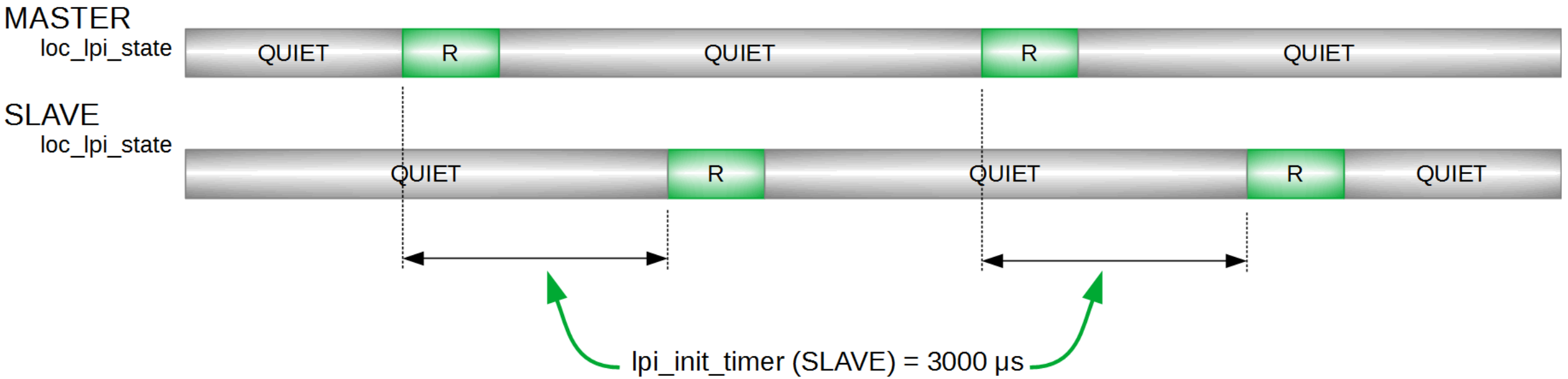


LPI QUIET REFRESH timing state diagram



- Timing offset between MASTER and SLAVE REFRESH is effected in LPI TIMER INIT state:
  - Define lpi_init_timer duration differently between MASTER and SLAVE
  - MASTER lpi_init_timer duration 0 µs
  - SLAVE lpi_init_timer duration 3000 µs
  - This offset, i.e. 3000 µs, should be maintained for lifetime of link
- Maintain lpi_refresh_timer and lpi_quiet_timer:
  - lpi_refresh_timer 250 µs
  - lpi_quiet_timer 6000 µs
- All timers here would be synchronized to symbol period (TX_TCLK). Might be defined in terms of symbol periods.
  - As SLAVE maintains timing lock with MASTER, so timing relationship between MASTER and SLAVE LPI QUIET REFRESH cycling should also remain fixed (and predictable)

LPI QUIET REFRESH cycling



- Time offset between MASTER and SLAVE LPI QUIET REFRESH cycling remains fixed for lifetime of link
  - 3000 µs, as per lpi_init_timer duration

PHY Control LPI sequencing

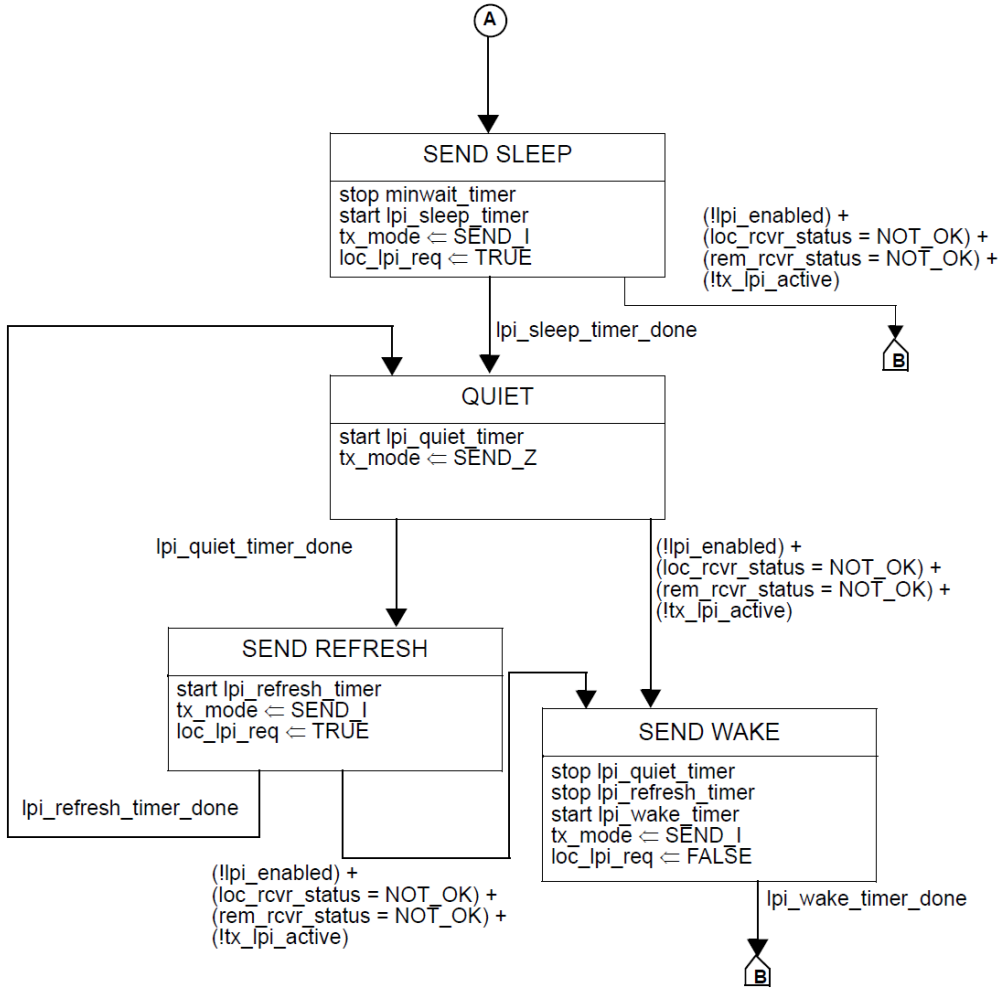
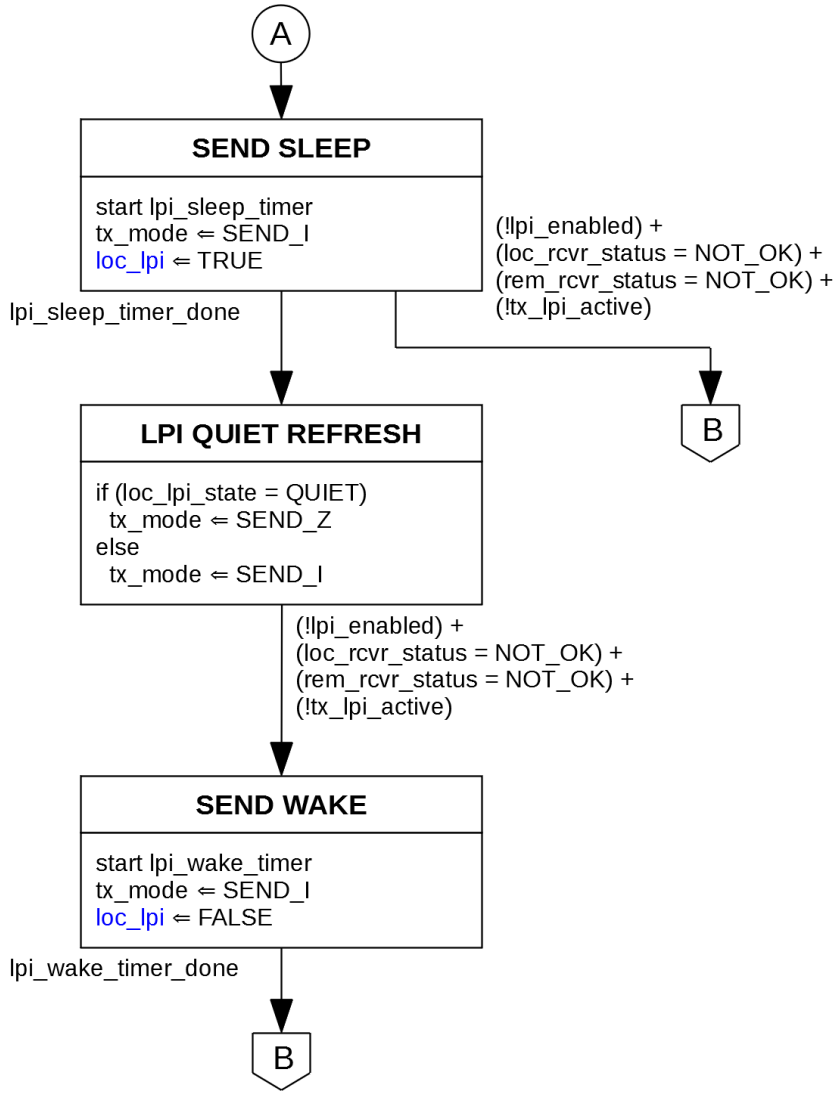
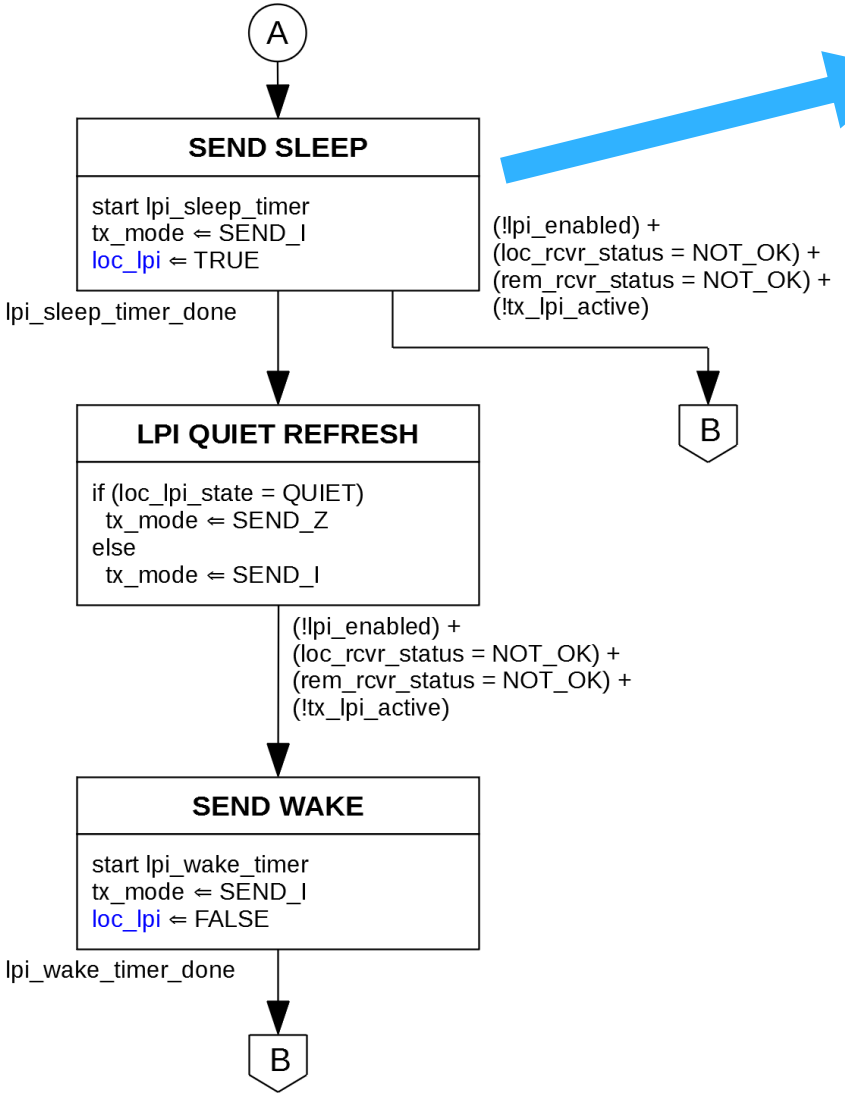


Figure 146–15—PHY Control state diagram (part b)

- PHY Control LPI sequencing is simplified
  - Combined LPI QUIET REFRESH state, with QUIET/REFRESH mode determined by loc_lpi_state variable

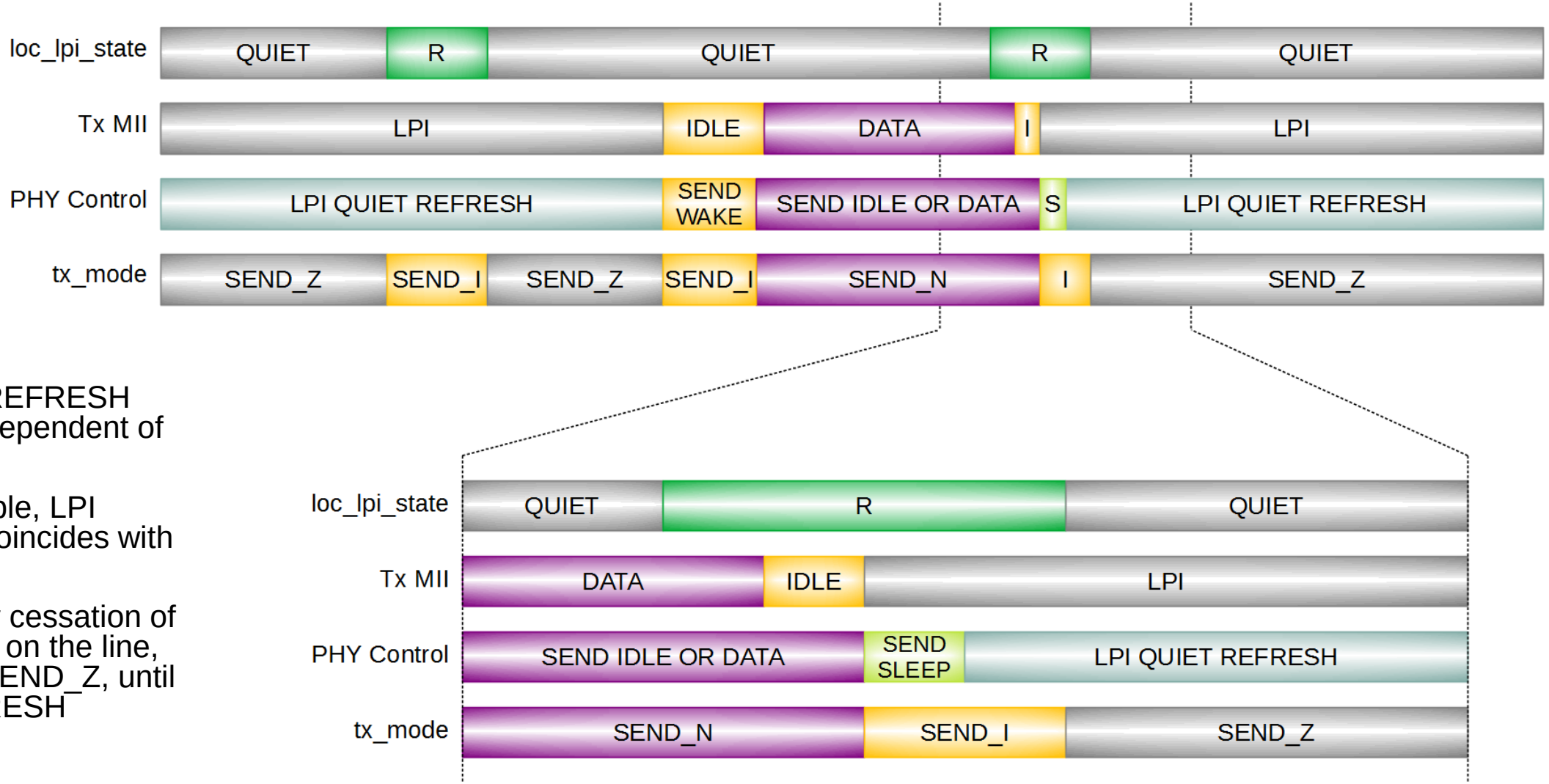


PHY Control LPI sequencing – lpi_sleep_timer duration



- Propose to reduce lpi_sleep_timer duration
  - Currently this is set to 250 µs (as per Graber_3cg_01a_0419)
  - The reason for the relatively long duration here is to allow any ongoing adaptation tasks to complete before transmission ceases in QUIET state.
  - But, given that link partner LPI QUIET REFRESH cycling can be known with certainty, a PHY should align to this, and should never require a longer duration in SEND SLEEP.
  - Propose lpi_sleep_timer duration of 20 µs, same as minwait_timer duration.
  - Might be specified in terms of transmit symbol periods.

LPI and frame transmission



- LPI QUIET REFRESH cycling is independent of data traffic
- In this example, LPI REFRESH coincides with end of frame
- Acts to delay cessation of transmission on the line, tx_mode = SEND_Z, until end of REFRESH

Potential energy savings

- The reduced duration of the lpi_sleep_timer provides for improved energy savings, as the QUIET state should be reached sooner than in the current draft.
  - A persistent data traffic pattern which provides a short period of LPI assertion between sending frames can even prevent entry to LPI QUIET altogether, if the period of LPI assertion is less than the lpi_sleep_timer. The only mitigation is to keep lpi_sleep_timer to a minimum.
- Knowledge of local and remote LPI refresh timing allows PHY implementation easier way of planning for filter coefficient updates

Thank you

Note on LPI signal naming

- loc_lpi_req and rem_lpi_req naming used in D3.0.
- These names have been lifted from 1000BASE-T (Clause 40), and are not used in other PHY standards.
- The naming is appropriate for the symmetric LPI scheme of 1000BASE-T, where the PHYs must both simultaneously signal a request for LPI mode in order for the LPI mode to be entered.
  - Where this is signalled from one PHY only it is only a request for LPI mode.
- In the context of 10BASE-T1L asymmetric LPI these names are inappropriate. For 10BASE-T1L, loc_lpi_req does not represent a request, and is rather a signal that the PHY transmit has entered the LPI mode of operation.

- Therefore, the following naming is proposed:
  - loc_lpi_req → loc_lpi
  - rem_lpi_req → rem_lpi