1000BASE-T PHY Control State Diagram Modifications

Niall Fitzgerald, Adam Healey,
Brian Murray, Jacobo Riesco
LSI Corporation

Presented by Adam Healey
IEEE P802.3az Task Force Meeting
New Orleans, LA
January 2009

Comments #98 and #101: Problem statement

- PHY Control state diagram allows a transition from UPDATE to WAKE to be forced at any time by the assertion of loc_lpi_req = FALSE
  - This results in continued transmission for lpi_waketx_timer followed by a period of silence (tx_mode = SEND_Z) no less than lpi_wakemz_timer

- This implies that the link partner's update of timing and adaptive filter coefficients could be interrupted at any time
  - Additional implementation constraint

- This permits pathological timing scenarios where LP_IDLE is asserted at the GMII such that the PHYs transitions to the UPDATE state and then LP_IDLE is de-asserted forcing the link partner to abort update of timing and adaptive filter coefficients
  - Repetitions of this timing cycle can starve the PHY of essential updates and degrade link performance
  - This issue could also be addressed by enforcing a minimum period that the "LPI client" must assert LP_IDLE

Approaches to comments #98 and #101 – 1

- Define the minimum time the "LPI client" must assert LP_IDLE
  - To ensure that both the local device and link partner both enjoy a period of uninterrupted transmission of a least lpi_update_timer (T_u)
  - No less than $2T_u(\text{min.}) + T_w(\text{min.})$, where T_w corresponds to lpi_wake_timer
  - No less than 376 microseconds
  - This translates directly to the size of the buffer that must be maintained by the transmitter

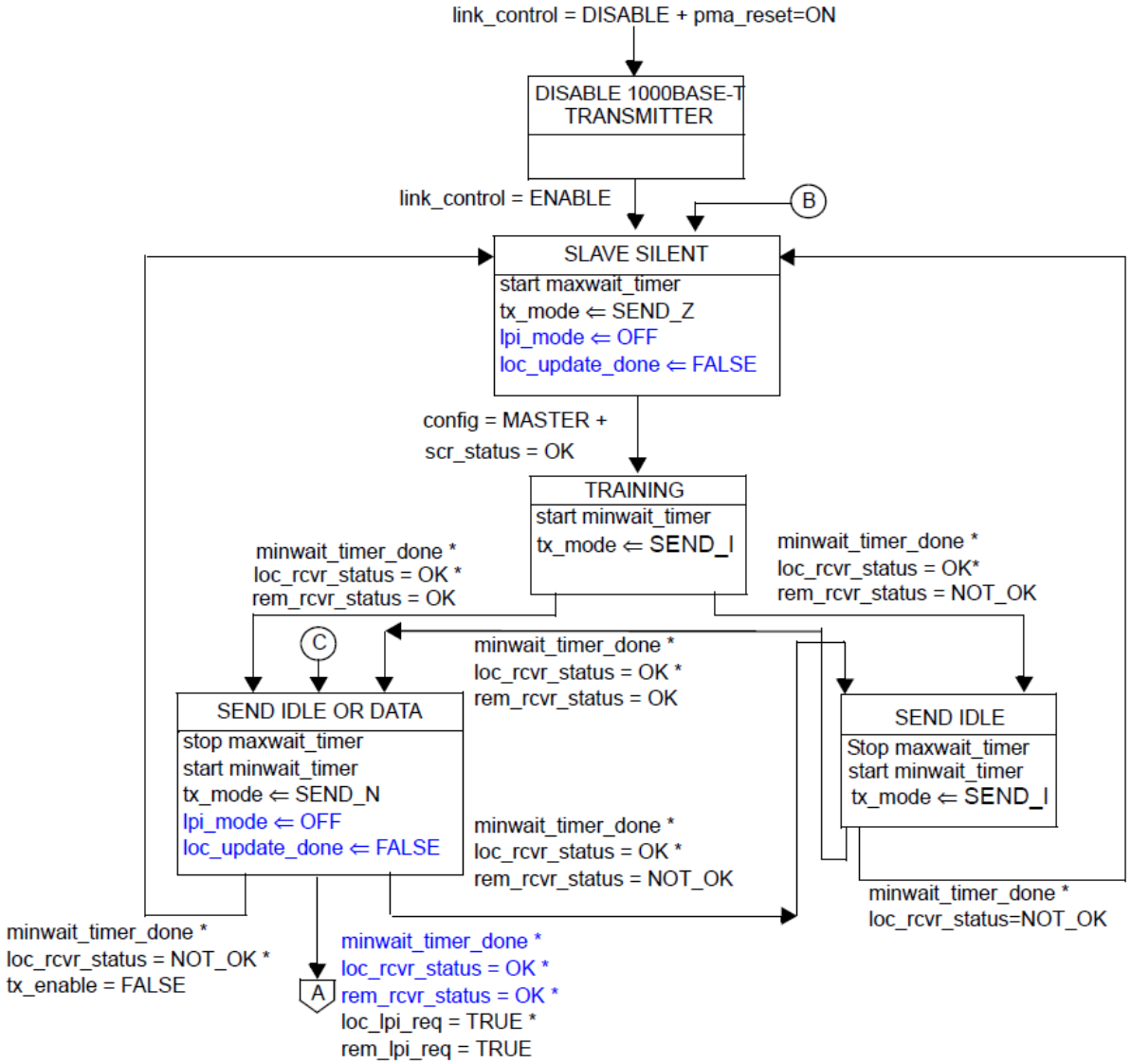
- Define the minimum time the client must wait between de-asserting LP_IDLE and asserting LP_IDLE again
  - Again, to ensure a period lpi_update_timer of uninterrupted transmission
  - No less than the minimum value of T_u (180 microseconds)

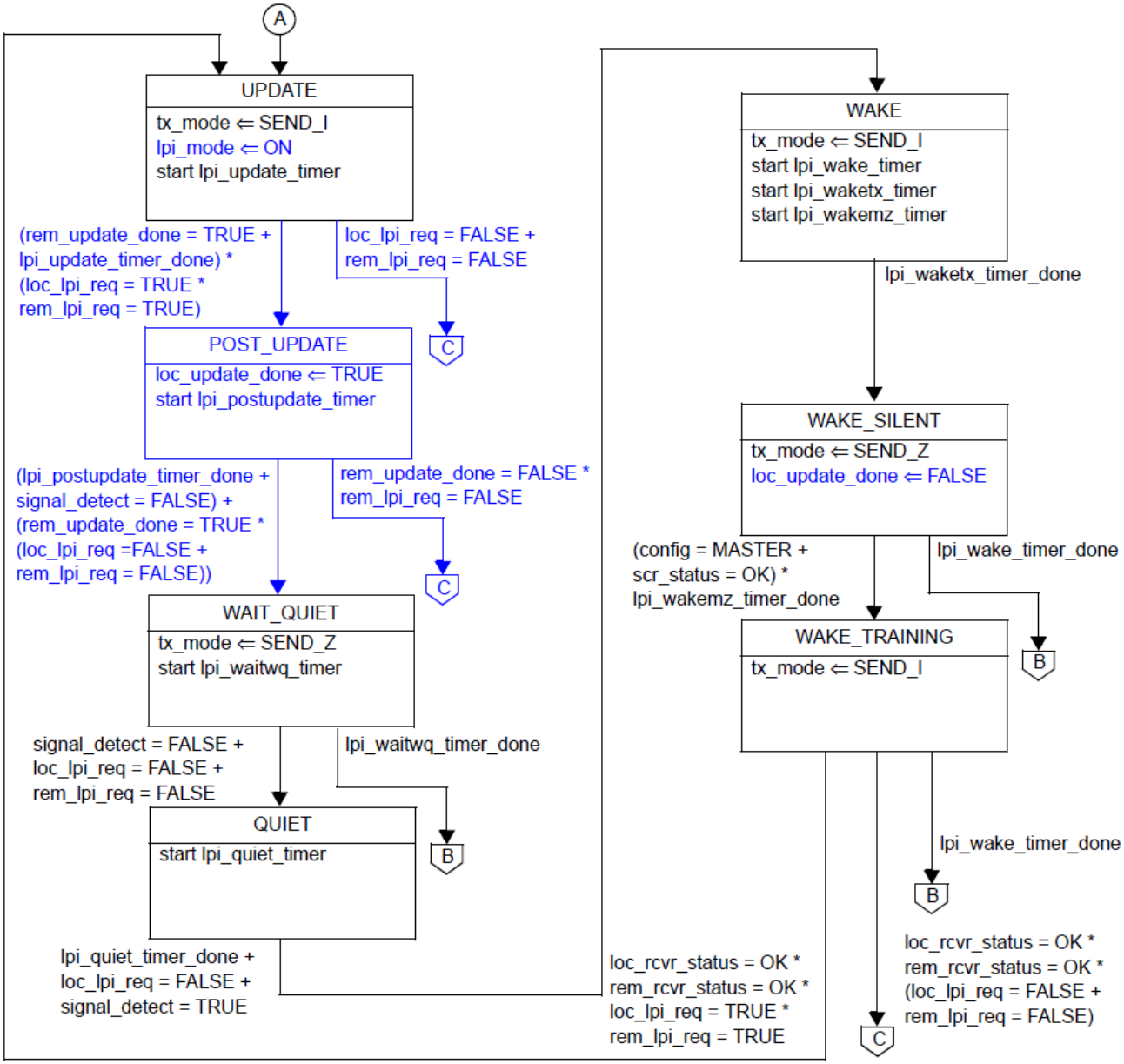
Approaches to comments #98 and #101 – 2

- However, these rules really address an issue with the 1000BASE-T PHY Control state diagram
  - Appropriate changes to the PHY Control state diagram would ensure proper operation of the PHY without any additional restrictions on the client
  - Avoid unwanted dependencies between proper operation of the client and proper operation of the PHY
  - Client does not need to make special provisions for a 1000BASE-T PHY
  - Address the root cause of the issue rather than consider work-arounds

Summary of proposed changes

- Introduce new POST_UPDATE state, succeeding the UPDATE state, controlling transitions into WAIT_QUIET or SEND IDLE OR DATA
- Restore lpi_mode to its Draft 1.0 definition
- Introduce new variable loc_update_done
  - Indicates completion of timing and adaptive filter coefficient updates
  - Assigned a value of FALSE prior to entering the UPDATE state
  - Assigned a value of TRUE in the POST_UPDATE state
  - Communicated to the link partner and received as rem_update_done
  - Use the same encoding rules adopted for loc_lpi_mode (possibly modified by comment #100)
- Remove the transition from WAKE_TRAINING to WAKE_SILENT
  - It was added to combat a fall-through case in the Draft 1.0 state diagram which no longer exists in Draft 1.1
- Remove lpi_waitwt_timer
  - It was added to combat a fall-through case in the Draft 1.0 state diagram which no longer exists in Draft 1.1


link_control = DISABLE + pma_reset=ON
DISABLE 1000BASE-T TRANSMITTER
link_control = ENABLE
B
SLAVE SILENT
start maxwait_timer
tx_mode ⇐ SEND_Z
lpi_mode ⇐ OFF
loc_update_done ⇐ FALSE
config = MASTER +
scr_status = OK
TRAINING
start minwait_timer
tx_mode ⇐ SEND_I
minwait_timer_done *
loc_rcvr_status = OK *
rem_rcvr_status = OK
minwait_timer_done *
loc_rcvr_status = OK*
rem_rcvr_status = NOT_OK
C
minwait_timer_done *
loc_rcvr_status = OK *
rem_rcvr_status = OK
SEND IDLE OR DATA
stop maxwait_timer
start minwait_timer
tx_mode ⇐ SEND_N
lpi_mode ⇐ OFF
loc_update_done ⇐ FALSE
SEND IDLE
Stop maxwait_timer
start minwait_timer
tx_mode ⇐ SEND_I
minwait_timer_done *
loc_rcvr_status = OK *
rem_rcvr_status = NOT_OK
minwait_timer_done *
loc_rcvr_status=NOT_OK
minwait_timer_done *
loc_rcvr_status = NOT_OK *
tx_enable = FALSE
A
minwait_timer_done *
loc_rcvr_status = OK *
rem_rcvr_status = OK *
loc_lpi_req = TRUE *
rem_lpi_req = TRUE


A
UPDATE
tx_mode ⇐ SEND_I
lpi_mode ⇐ ON
start lpi_update_timer
(rem_update_done = TRUE +
lpi_update_timer_done) *
(loc_lpi_req = TRUE *
rem_lpi_req = TRUE)
loc_lpi_req = FALSE +
rem_lpi_req = FALSE
C
POST_UPDATE
loc_update_done ⇐ TRUE
start lpi_postupdate_timer
(lpi_postupdate_timer_done +
signal_detect = FALSE) +
(rem_update_done = TRUE *
(loc_lpi_req =FALSE +
rem_lpi_req = FALSE))
rem_update_done = FALSE *
rem_lpi_req = FALSE
C
WAIT_QUIET
tx_mode ⇐ SEND_Z
start lpi_waitwq_timer
signal_detect = FALSE +
loc_lpi_req = FALSE +
rem_lpi_req = FALSE
lpi_waitwq_timer_done
QUIET
start lpi_quiet_timer
B
lpi_quiet_timer_done +
loc_lpi_req = FALSE +
signal_detect = TRUE
WAKE
tx_mode ⇐ SEND_I
start lpi_wake_timer
start lpi_waketx_timer
start lpi_wakemz_timer
lpi_waketx_timer_done
WAKE_SILENT
tx_mode ⇐ SEND_Z
loc_update_done ⇐ FALSE
(config = MASTER +
scr_status = OK) *
lpi_wakemz_timer_done
lpi_wake_timer_done
WAKE_TRAINING
tx_mode ⇐ SEND_I
B
lpi_wake_timer_done
B
loc_rcvr_status = OK *
rem_rcvr_status = OK *
(loc_lpi_req = FALSE +
rem_lpi_req = FALSE)
C
loc_rcvr_status = OK *
rem_rcvr_status = OK *
loc_lpi_req = TRUE *
rem_lpi_req = TRUE

Highlights

- A direct transition is provided from UPDATE (or POST_UPDATE) to SEND IDLE OR DATA if the link partner has not yet completed filter coefficient updates (e.g. rem_update_done = FALSE)
  - Update of adaptive filter coefficients may continue uninterrupted
- When the remote PHY has signaled completion of update then the transition through to the wake sequence is possible
- Duration of lpi_postupdate_timer is required to be greater than one round-trip delay
  - Propose a range of 2.0 and 2.2 microseconds
- If loc_lpi_req = FALSE during POST_UPDATE, then the local device must wait for rem_update_done = TRUE before proceeding to WAKE
  - This will not add time to the overall wake time budget

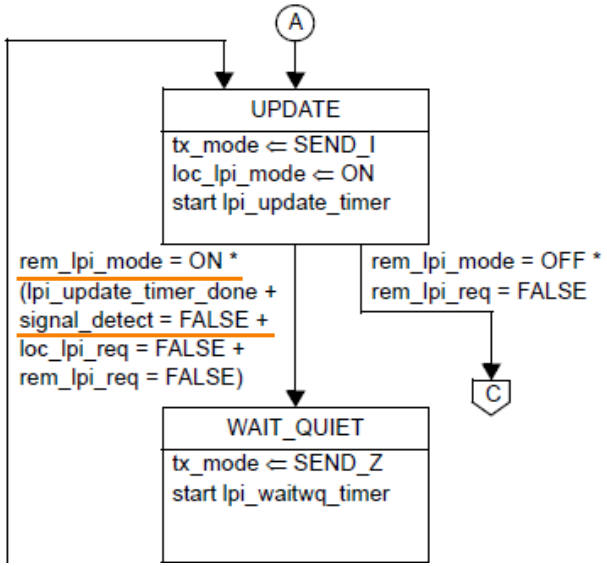
Notes on comments #12 and #87

- Proposed state diagram also addresses corner cases pointed out in comments #12 and #87
- It does not get stuck in the UPDATE state
- There are no ambiguous state transitions
  - Decoded variables rem_lpi_req and rem_lpi_mode are not AND'ed with the condition signal_detect = FALSE
  - Requires one to assume default values for rem_lpi_req and rem_lpi_mode since they cannot be derived
- It does not suffer from the "out-of-sync" condition possible from the Draft 1.1 state diagram

Ambiguous transition in Draft 1.1 state diagram

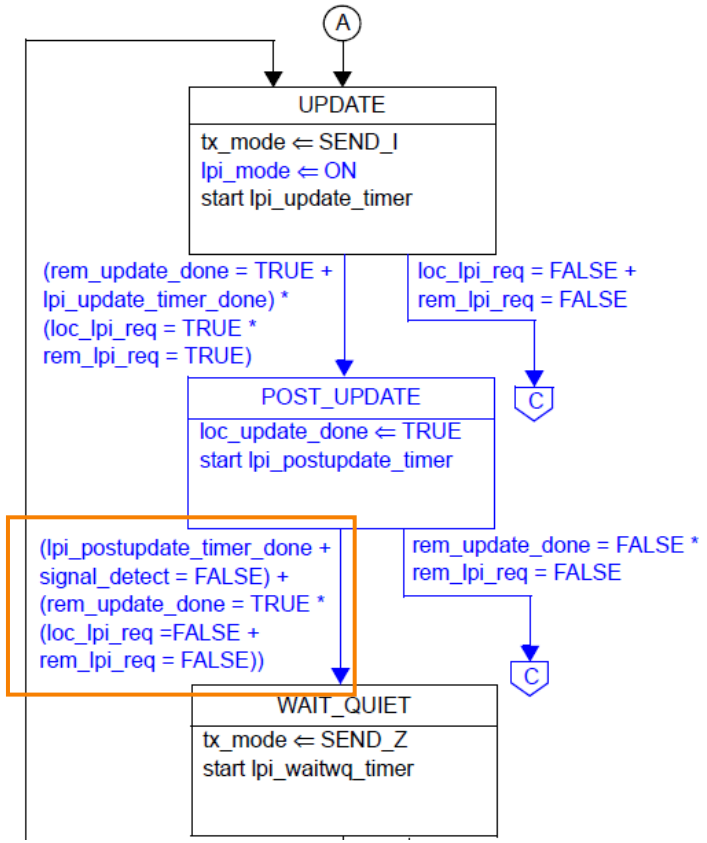
- Transition condition from UPDATE to WAIT_QUIET includes the term rem_lpi_mode = ON * signal_detect = FALSE

- How can rem_lpi_mode be detected when there is no signal present?
  - What value of rem_lpi_mode should be assumed?

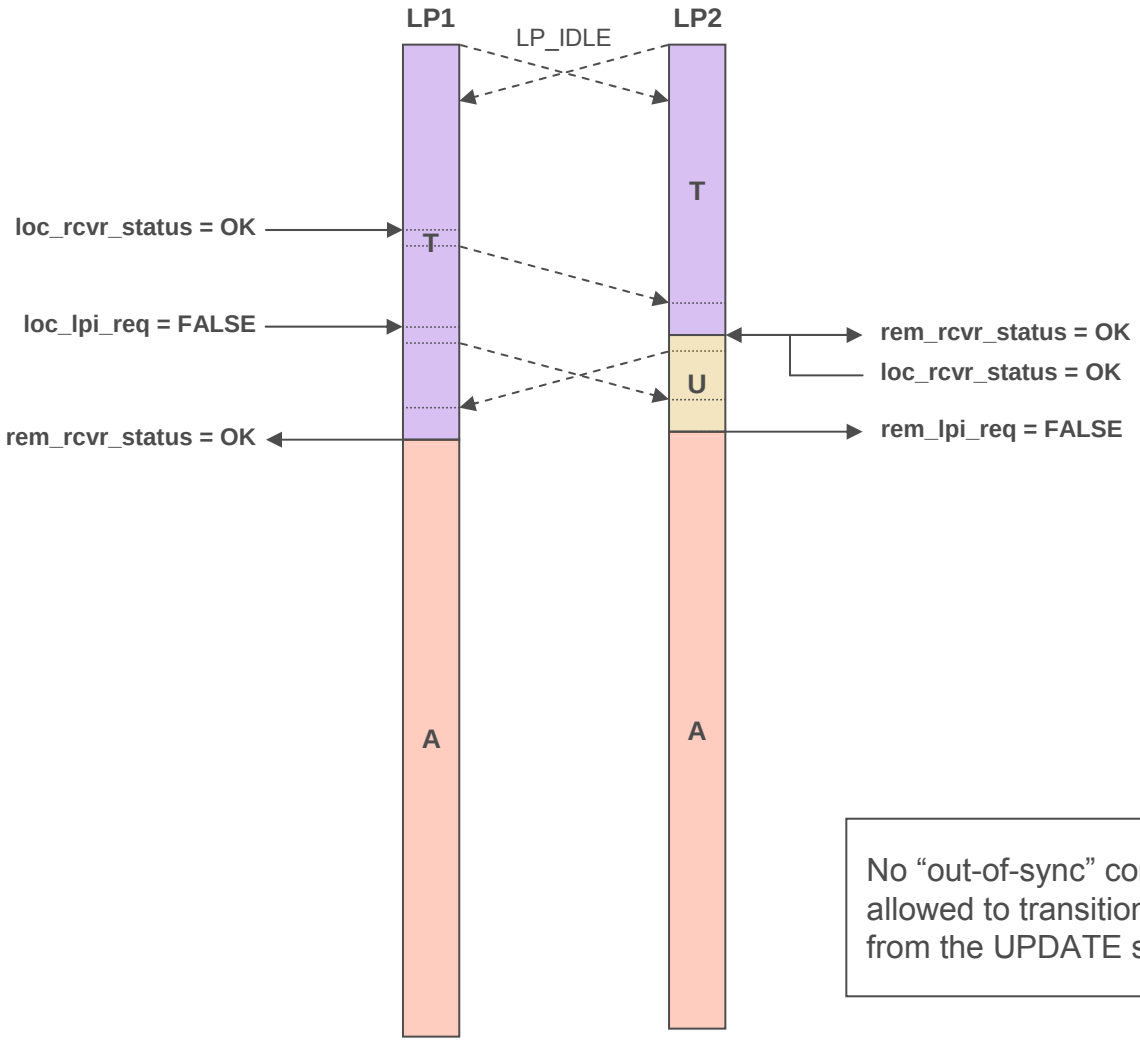


Comment #12: Stuck in update

- Condition could modified to remove this ambiguity
- signal_detect = FALSE + lpi_update_timer_done + rem_lpi_mode = ON * (loc_lpi_req = FALSE + rem_lpi_req = FALSE)
- Correction eliminates corner cases pointed out in Comment #12
- Correction already presented in the proposed state diagram
  - POST_UPDATE to WAIT_QUIET



Comment #87: Corrected by proposed state diagram



LEGEND

- A — ACTIVE (SEND IDLE OR DATA)
- U — UPDATE
- P — POST_UPDATE
- WQ — WAIT_QUIET
- Q — QUIET
- W — WAKE
- Z — WAKE_SILENT
- T — WAKE_TRAINING

No “out-of-sync” condition occurs because the PHY is allowed to transition to the SEND IDLE OR DATA state from the UPDATE state

Comments #98 and #101: Summary

- Proposed state diagram addresses comments #12, #87, #98, and #101

- Eliminates a timer and state transition that were made obsolete by changes from Draft 1.0 to Draft 1.1.

Comment #102: Problem statement – 1

- What are the consequences of not completing wake within the allotted time?
  - Completion of wake is defined as the point when both loc_rcvr_status and rem_rcvr_status are OK

- Packet(s) transmitted immediately after lpi_wake_timer_done are lost
  - tx_mode ≠ SEND_N until wake is complete, inhibiting the transmission of frames
  - For this reason, it is imperative to set PHY parameters so that the chances of failing to complete wake within the allotted time are very small
  - For all practical purposes, the probability of failing to wake should be less than the target probability of packet error

- During a refresh or when system wake time greatly exceeds the PHY wake time, the consequences are minor
  - No data loss, perhaps a very small compromise of power savings

Comment #102: Problem statement – 2

- Failure to achieve both loc_rcvr_status = OK and rem_rcvr_status = OK prior to lpi_wake_timer_done causes PHY Control to transition to the SLAVE SILENT state and initiate re-training
  - This will correspond to an interruption of service spanning hundreds of milliseconds
- Consequences are considerably more severe in all cases when re-training is enforced
- Even though such event should be rare, the same could be said of frequency of a packet error during normal operation
  - During normal operation, the link is not restarted for every packet error

Comment #102: Summary of proposal

- Use lpi_wake_timer to monitor the health of the link
  - Define that lpi_wake_timer_done causes a new counter, “1000BASE-T wake error,” to be incremented
  - Counter is represented in the Clause 45 management register space and is cleared on read
  - System management reads the counter to understand if the link is failing to recover from low-power mode within the allotted time and takes corrective actions as necessary

- Define a new timer, lpi_link_fail_timer
  - Functionally replaces lpi_wake_timer in the PHY Control state diagram, e.g. expiration triggers re-training
  - Started in the WAKE state
  - Propose timer value to be 90 to 110 microseconds

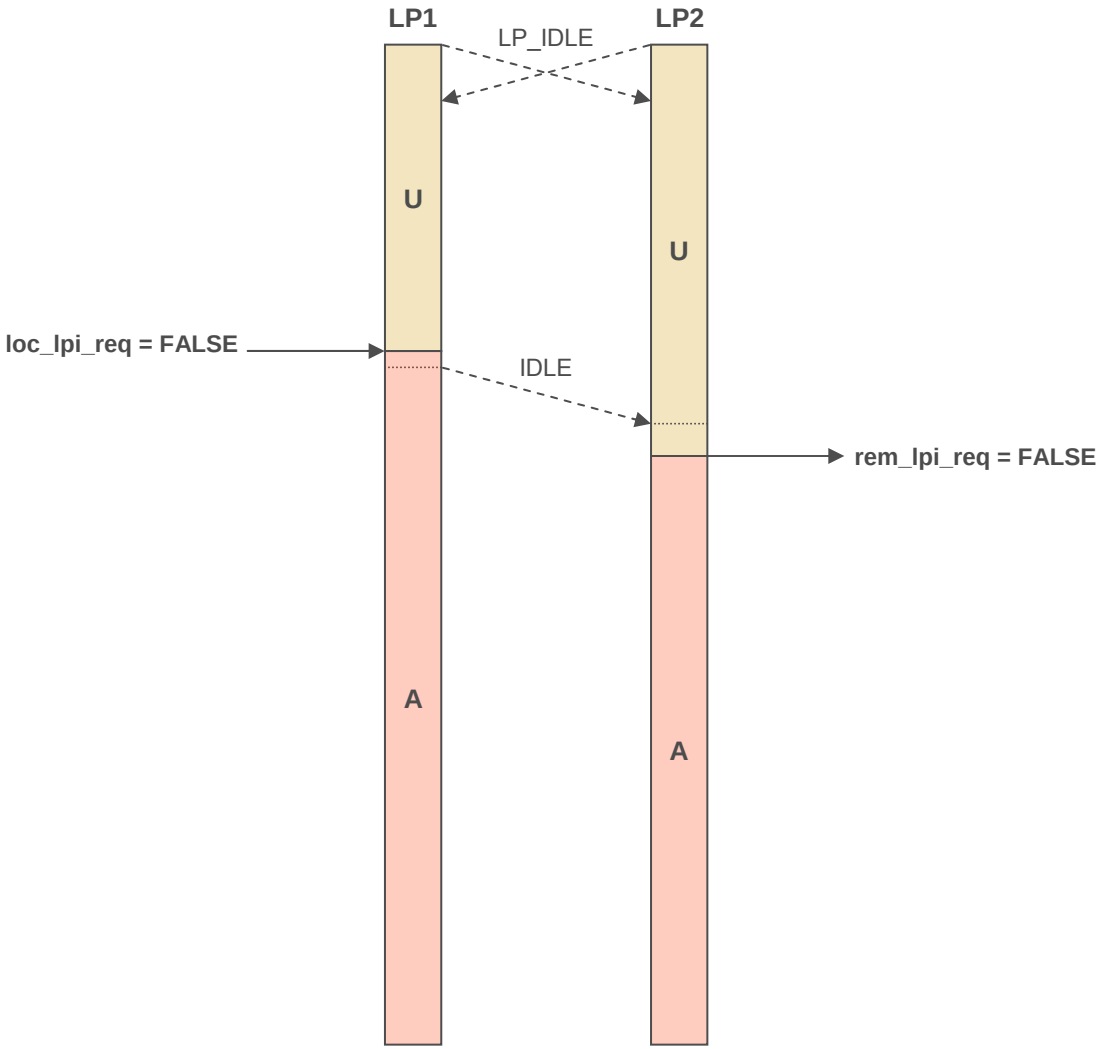
- Add action “Stop lpi_wake_timer” to SEND IDLE OR DATA to prevent lpi_wake_timer_done from being satisfied after completion of wake

LSI™

Questions?

Back-up slides

Wake from UPDATE

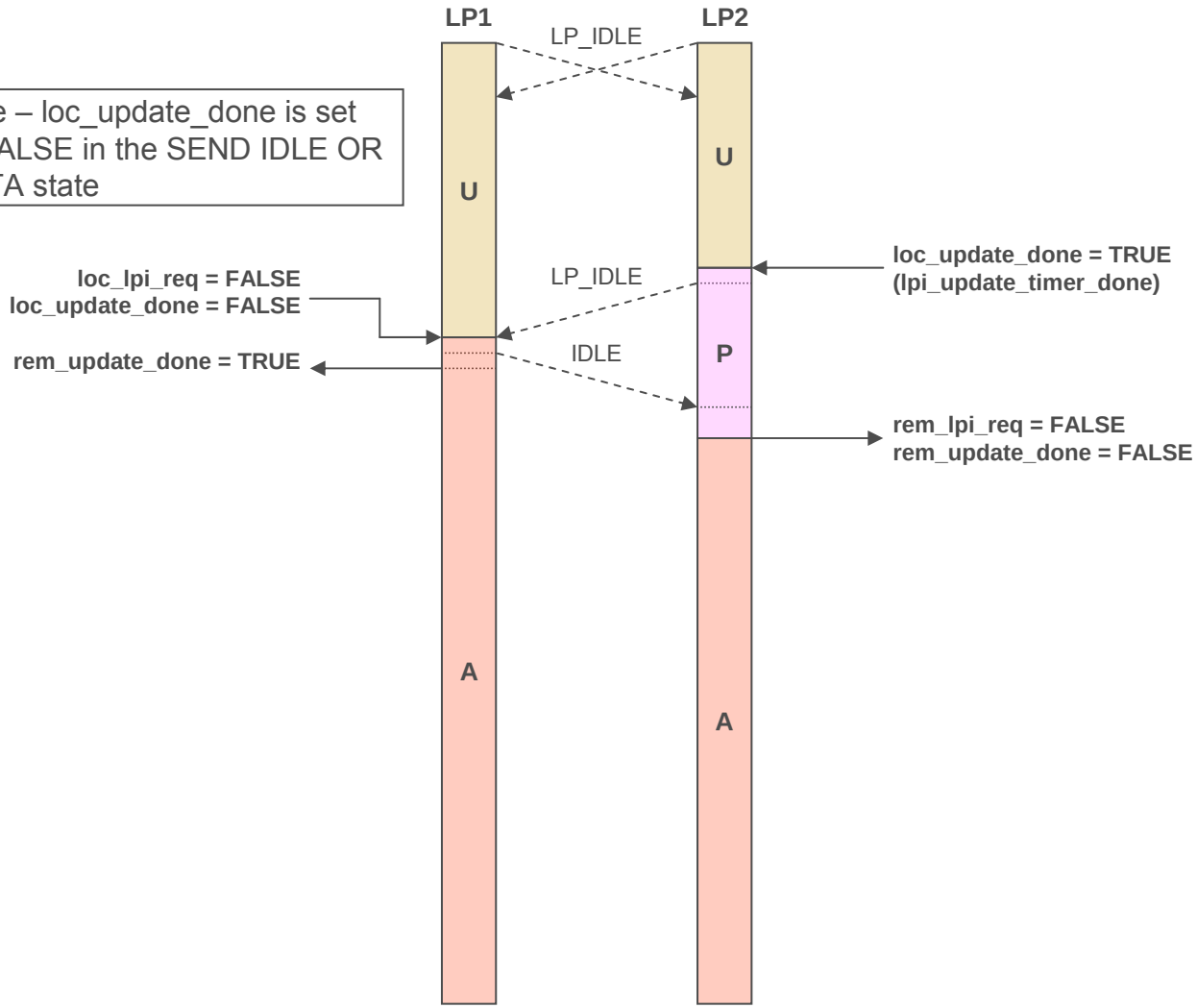


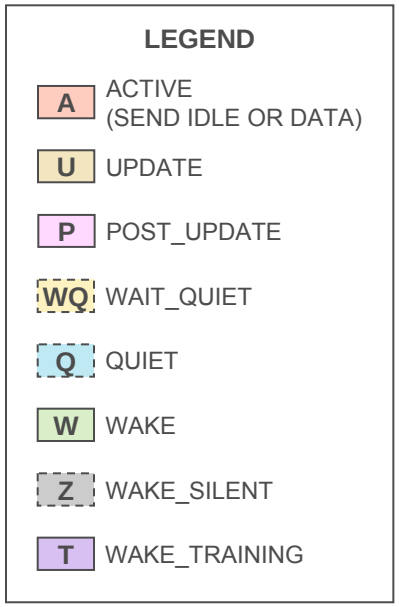
LEGEND

- A — ACTIVE (SEND IDLE OR DATA)
- U — UPDATE
- P — POST_UPDATE
- WQ — WAIT_QUIET
- Q — QUIET
- W — WAKE
- Z — WAKE_SILENT
- T — WAKE_TRAINING

Wake from POST_UPDATE – 1

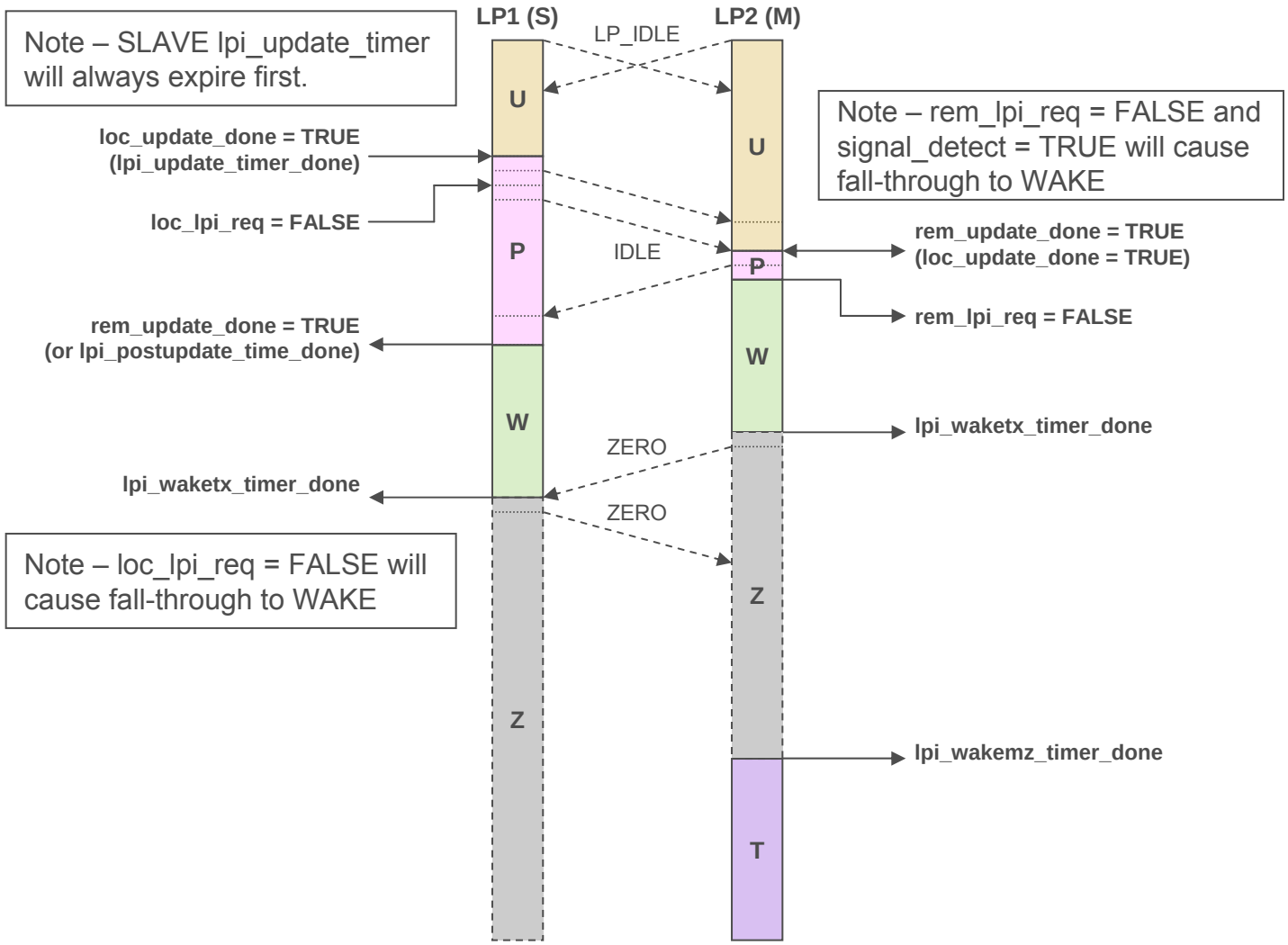
Note – loc_update_done is set to FALSE in the SEND IDLE OR DATA state





Wake from POST_UPDATE – 2

Note – SLAVE lpi_update_timer will always expire first.



Note – rem_lpi_req = FALSE and signal_detect = TRUE will cause fall-through to WAKE

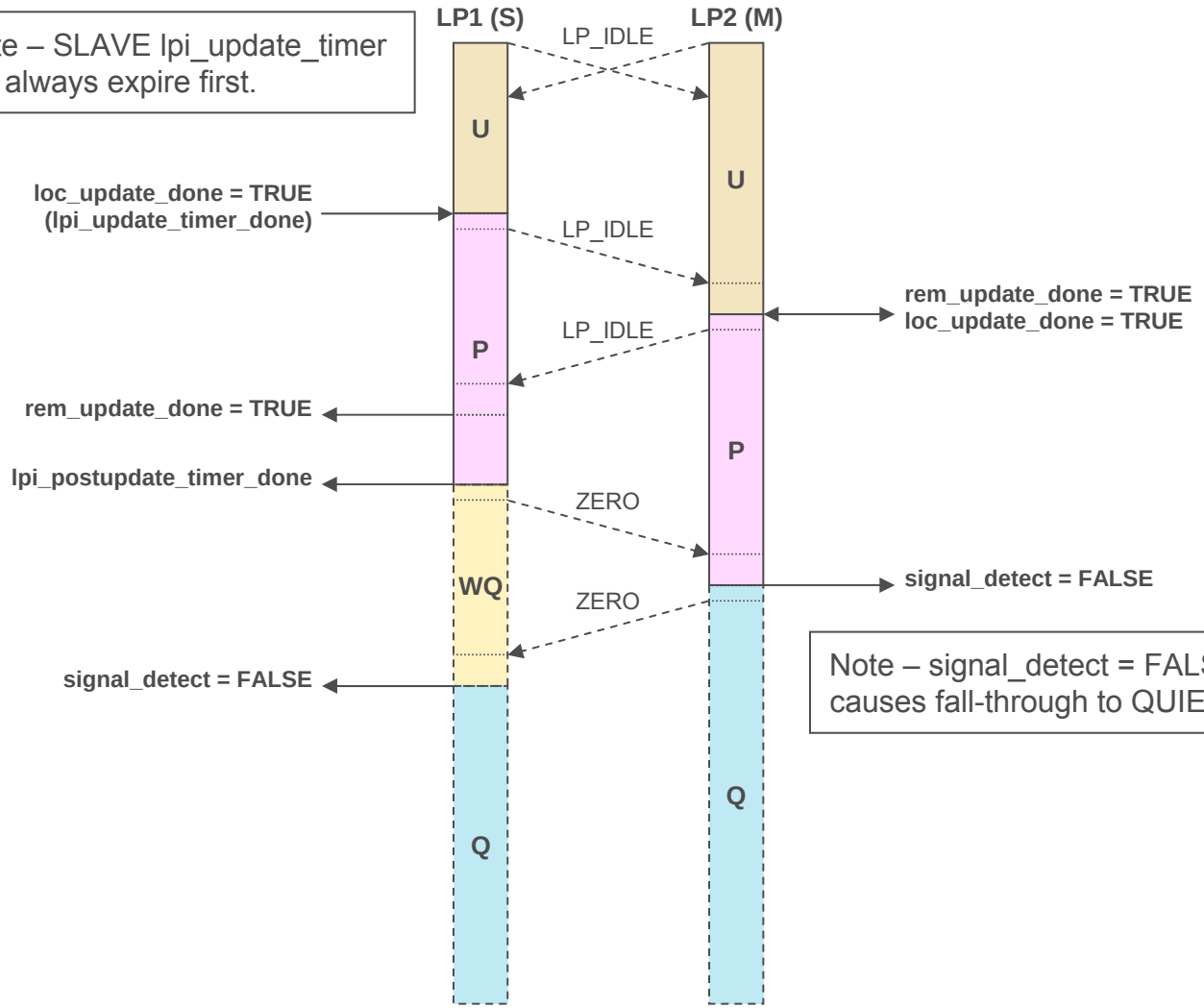
Note – loc_lpi_req = FALSE will cause fall-through to WAKE

LEGEND

- A — ACTIVE (SEND IDLE OR DATA)
- U — UPDATE
- P — POST_UPDATE
- WQ — WAIT_QUIET
- Q — QUIET
- W — WAKE
- Z — WAKE_SILENT
- T — WAKE_TRAINING

Enter QUIET from POST_UPDATE

Note – SLAVE lpi_update_timer will always expire first.

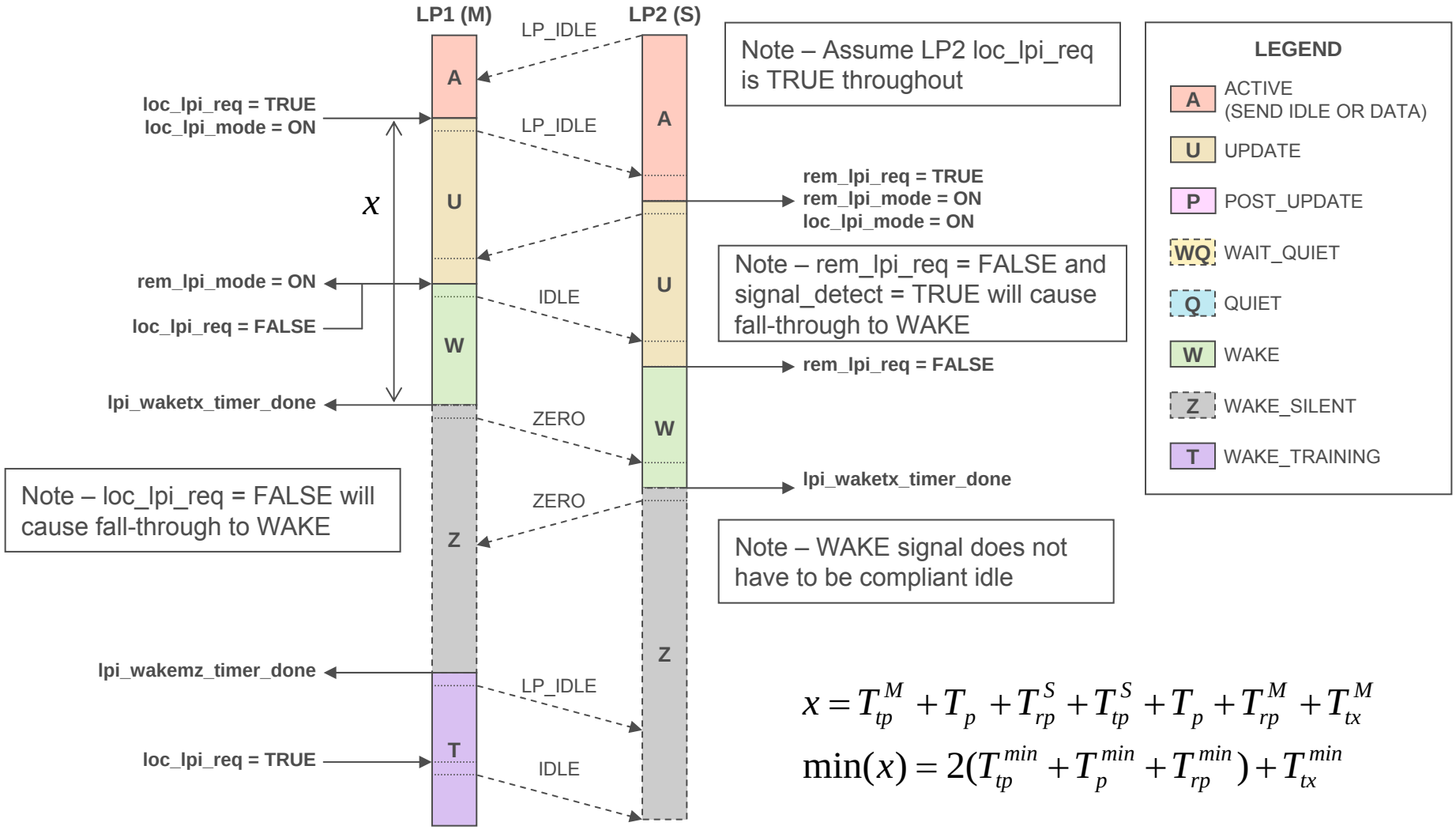


Note – signal_detect = FALSE causes fall-through to QUIET.

LEGEND

- A – ACTIVE (SEND IDLE OR DATA)
- U – UPDATE
- P – POST_UPDATE
- WQ – WAIT_QUIET
- Q – QUIET
- W – WAKE
- Z – WAKE_SILENT
- T – WAKE_TRAINING

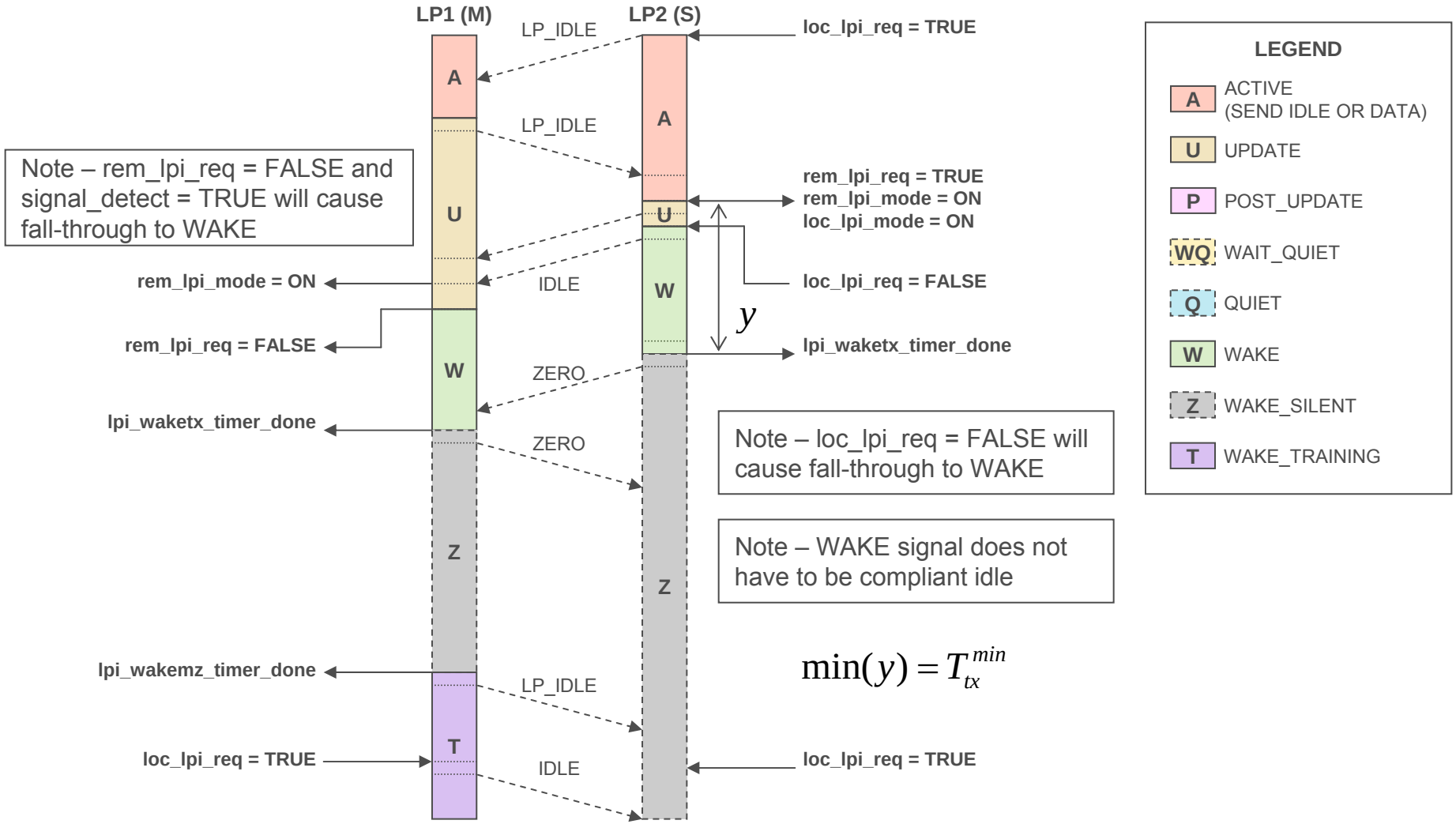
Update starvation: Draft 1.1 timing analysis – 1



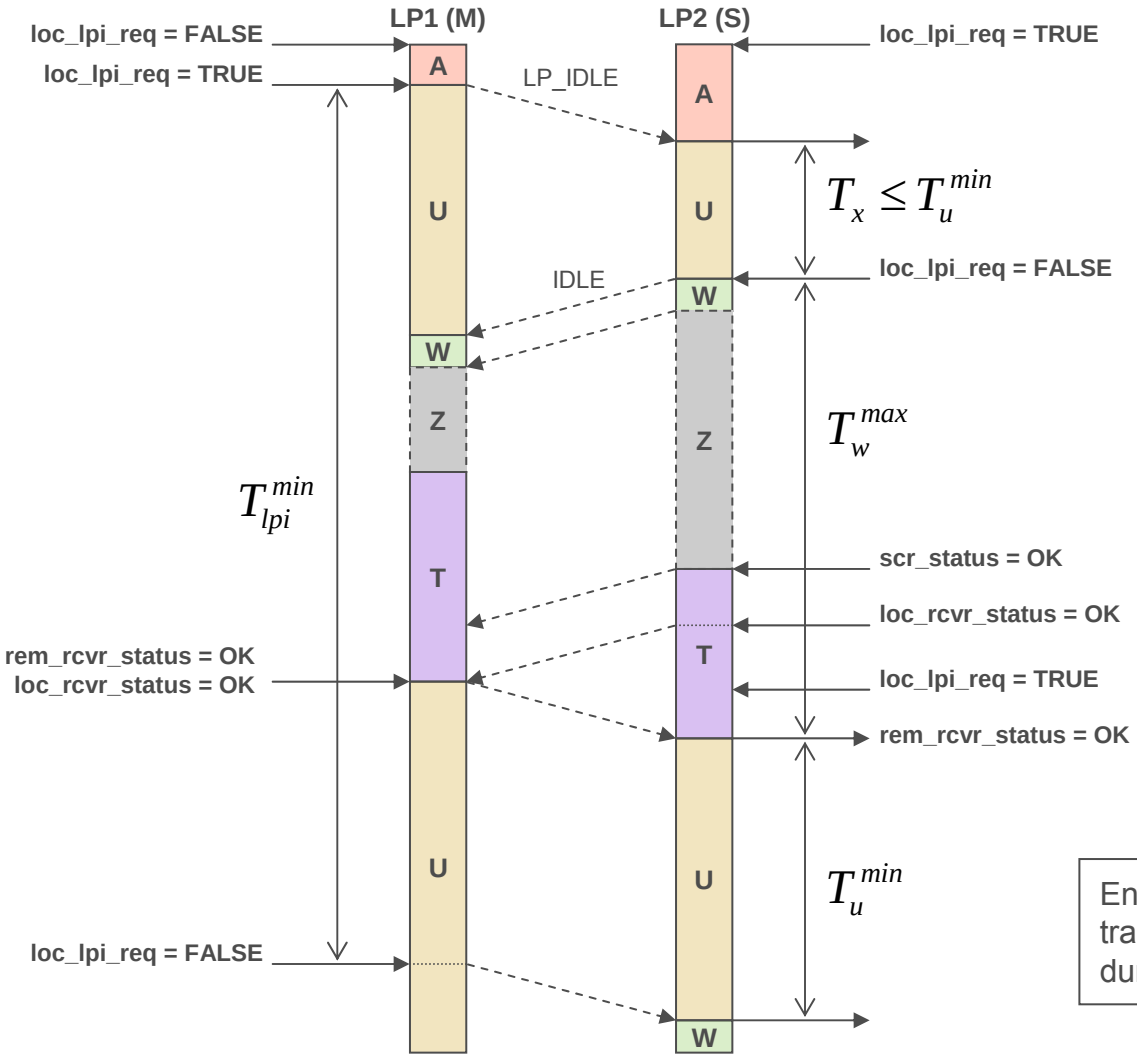
$$x = T_{tp}^{M} + T_{p} + T_{rp}^{S} + T_{tp}^{S} + T_{p} + T_{rp}^{M} + T_{tx}^{M}$$

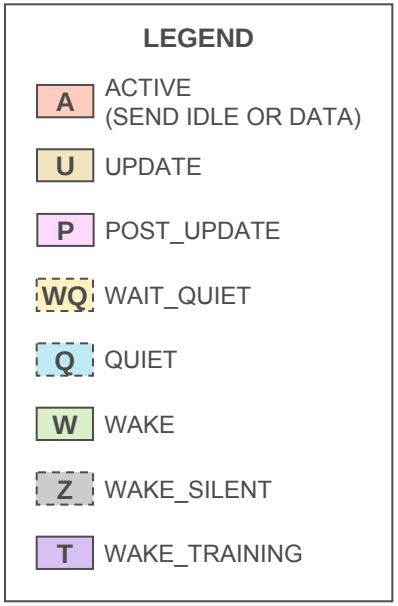
$$\min(x) = 2(T_{tp}^{min} + T_{p}^{min} + T_{rp}^{min}) + T_{tx}^{min}$$

Update starvation: Draft 1.1 timing analysis – 2



Update starvation: Draft 1.1 timing analysis – 3





$$T_{lpi}^{min} \geq 2T_u^{min} + T_w^{max}$$

Ensures a period of continuous transmission at least $T_u(min.)$ in duration