Corner cases and Comments on EEE Clause 40

Sep 2008
IEEE 802.3az Task Force

Joseph Chou

Realtek Semiconductor Corp.

Comment 1: SEND_Z won't send Zero

- loc_rcvr_status = FALSE in WAIT_QUIET , QUIET , and WAKE_SILENT states
  ➔ channel C output will be safely staying at zero value.
- However, during LPI quiet state, loc_lpi_req=TRUE makes Sdn[3]=Scn[3]^1
  ➔ channel D output will not stay at zero as desired (tx_mode=SEND_Z).
- Therefore, need to inverse Scn[3] (by EXOR with 1) only if loc_lpi_req=TRUE **and tx_mode≠SEND_Z**

$$Sd_n[2] = \begin{cases} Sc_n[2] \wedge TXD_n[2] & \text{if } (tx\_enable_{n-2} = 1) \\ Sc_n[2] \wedge 1 & \text{else if } (loc\_rcvr\_status = \text{OK}) \\ Sc_n[2] & \text{else} \end{cases}$$

$$Sd_n[3] = \begin{cases} Sc_n[3] \wedge TXD_n[3], & if\ (tx\_enable_{n-2} = 1) \\ Sc_n[3] \wedge 1, & else\ if\ (loc\_lpi\_req = TRUE) \\ Sc_n[3], & else \end{cases}$$

loc_lpi_req=TRUE * tx_mode≠SEND_Z

Comment 2: SEND_Z won't send Z （cont）

- tx_error is encoded in the channel B output Sdn(1)
- The opcode used for Carrier_extention (0x0F) is already excluded from the encoding
- By the same token, the opcode used for the LPI mode (0x01, TX_LP_IDLE) needs to be excluded from the encoding of Sdn(1) to avoid the non-zero output during SEND_Z.

$$Sd_n[1] = \begin{cases} Sc_n[1] \text{ ^}TXD_n[1], \text{ if}(tx\_enable_{n-2}=1) \\ Sc_n[1]\text{^ } cext\_err_n \text{ ,else} \end{cases}$$

$$cext\_err_n = \begin{cases} tx\_error_n \text{ if}(tx\_enable_n=0 \\ \quad *TXD_n[7:0]) \neq 0x0F \\ \quad *TXD_n[7:0]) \neq 0x01) \\ 0 \text{ else} \end{cases}$$

Comment 3: signal_detect

- Original text:

signal_detect
The signal_detect variable is set by the PMA Receive function and indicates the presence of a signal at the MDI.

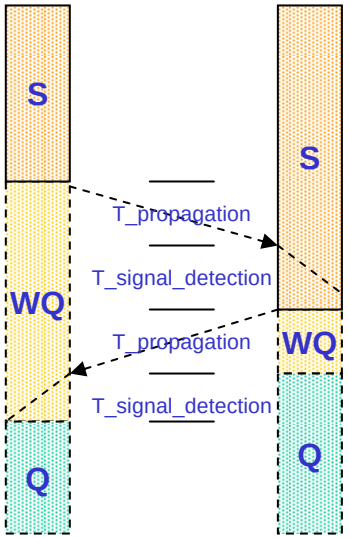
Values: TRUE: There is a signal present at the MDI.
FALSE: There is no signal present at the MDI.

- Need to define the timing constraint to be referred by various LPI timers (lpi_waitqt_timer, lpi_quietmin_timer, lpi_waketx_timer, lpi_waitact_timer, etc.)
- Propose a maximum value of **1 us** or smaller if feasible

Comment 4: separate lpi_wait_timer

- **Some intermediate LPI states need minimum time to transit and maximum time to break out**
- **Better to have separate timer for each state**

lpi_waitqt_timer
>>
2 * propagation delay +
2* signal_detection time
= 3.6us
Suggested value = 10us



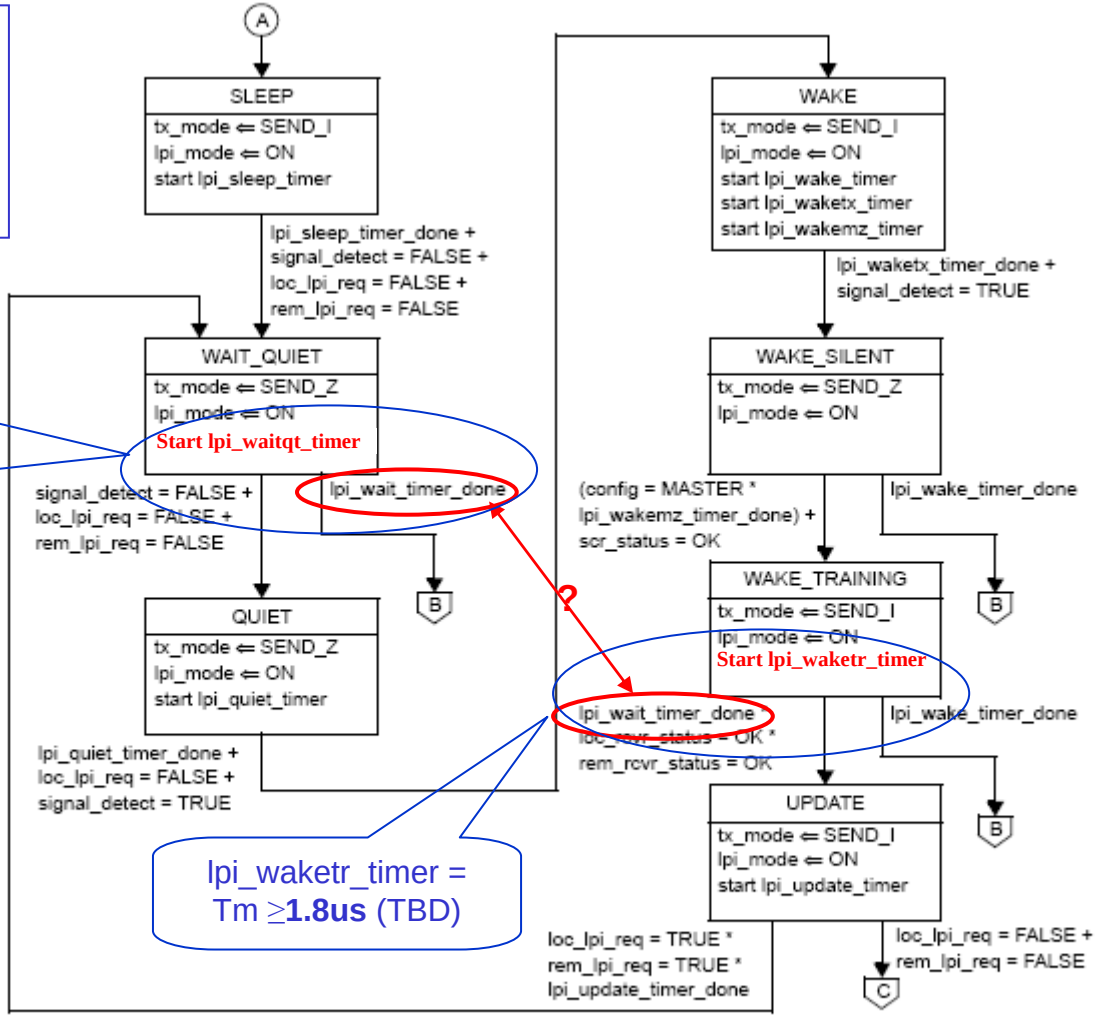
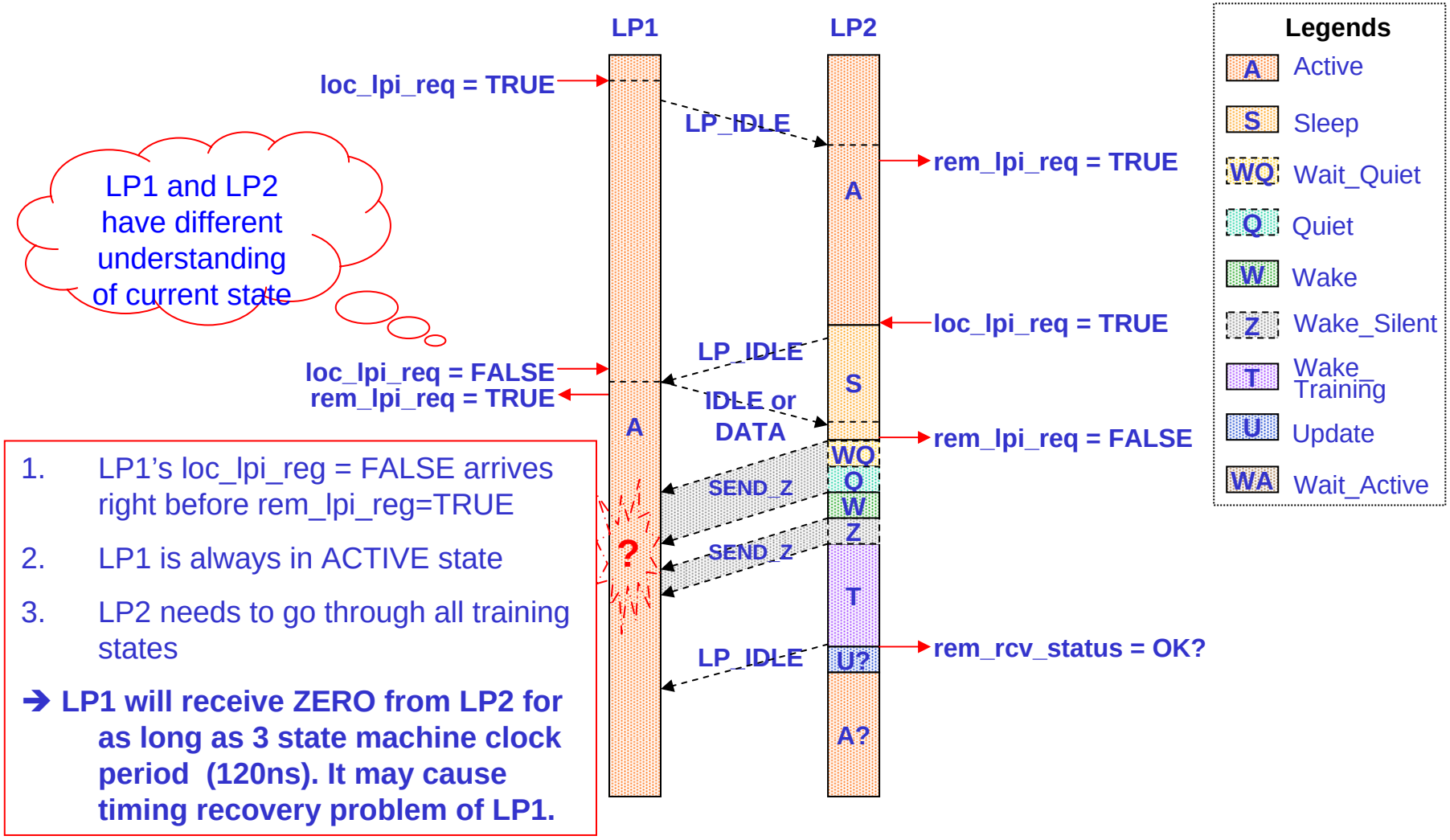


Figure 40–15b—PHY Control state diagram, part b

Comment 5: LPI states out of sync

- The spirit of EEE GPHY state transitions is to mimic the cold start of GPHY link so that the local and remote sites, either Master or Slave, can keep synchronization of training states to set up the appropriate CDR loop.
- However, either party of an EEE link segment can exit any intermediate state and accelerate the transition of rest of states resulting in a temporary out of sync of state coherency between two sides due to the round trip delay of signal exchanged.
- During the acceleration of state transition, the unwanted SEND_Z (WAIT_QUIET, QUIET, WAKE_SILENT) output may cause the timing loop failure and mess up the descrambler and decoder.

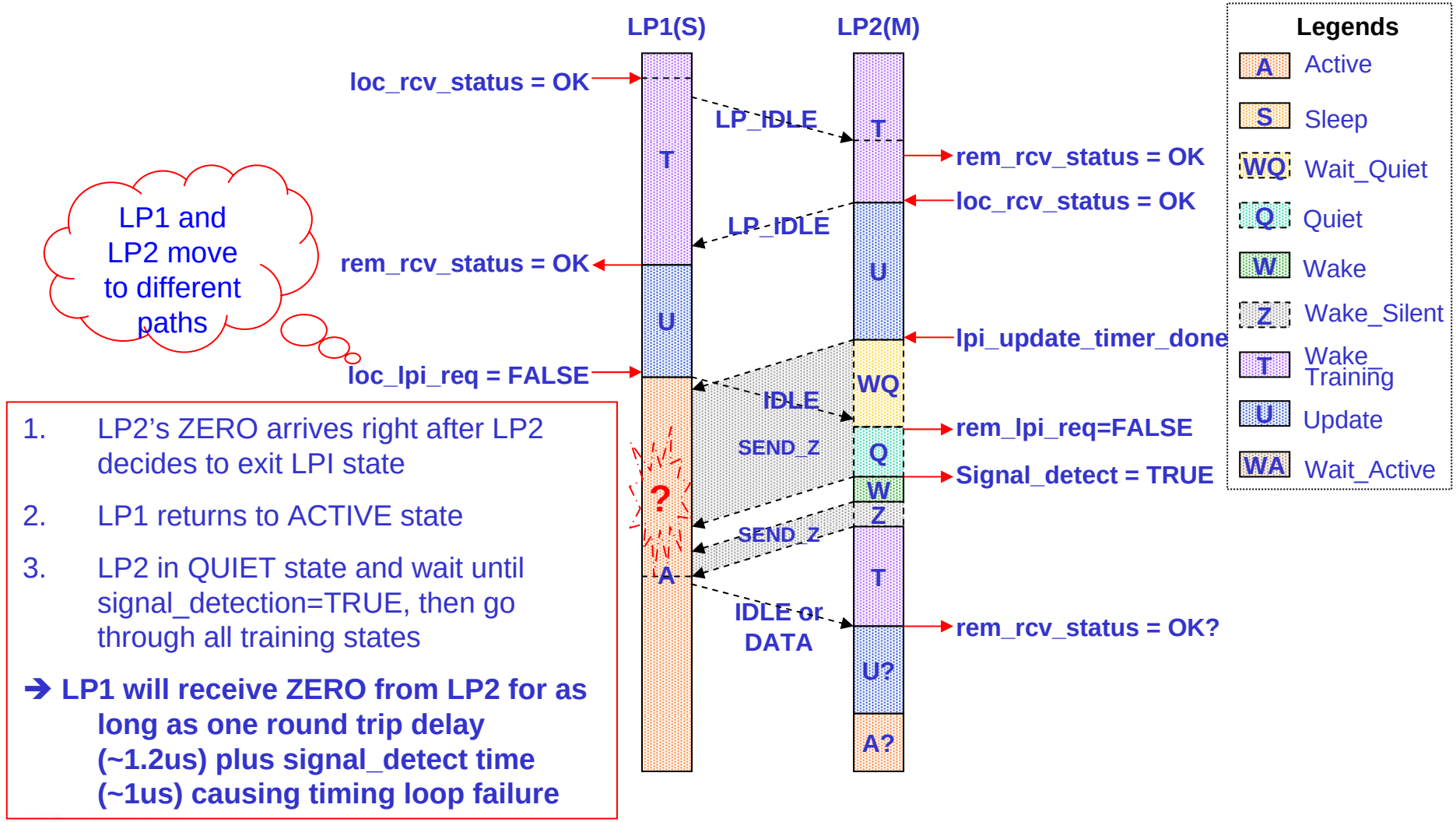
Case 1: LP1 is always Active, LP2 goes through all transition states



1. LP1's loc_lpi_reg = FALSE arrives right before rem_lpi_reg=TRUE
2. LP1 is always in ACTIVE state
3. LP2 needs to go through all training states

➔ **LP1 will receive ZERO from LP2 for as long as 3 state machine clock period (120ns). It may cause timing recovery problem of LP1.**

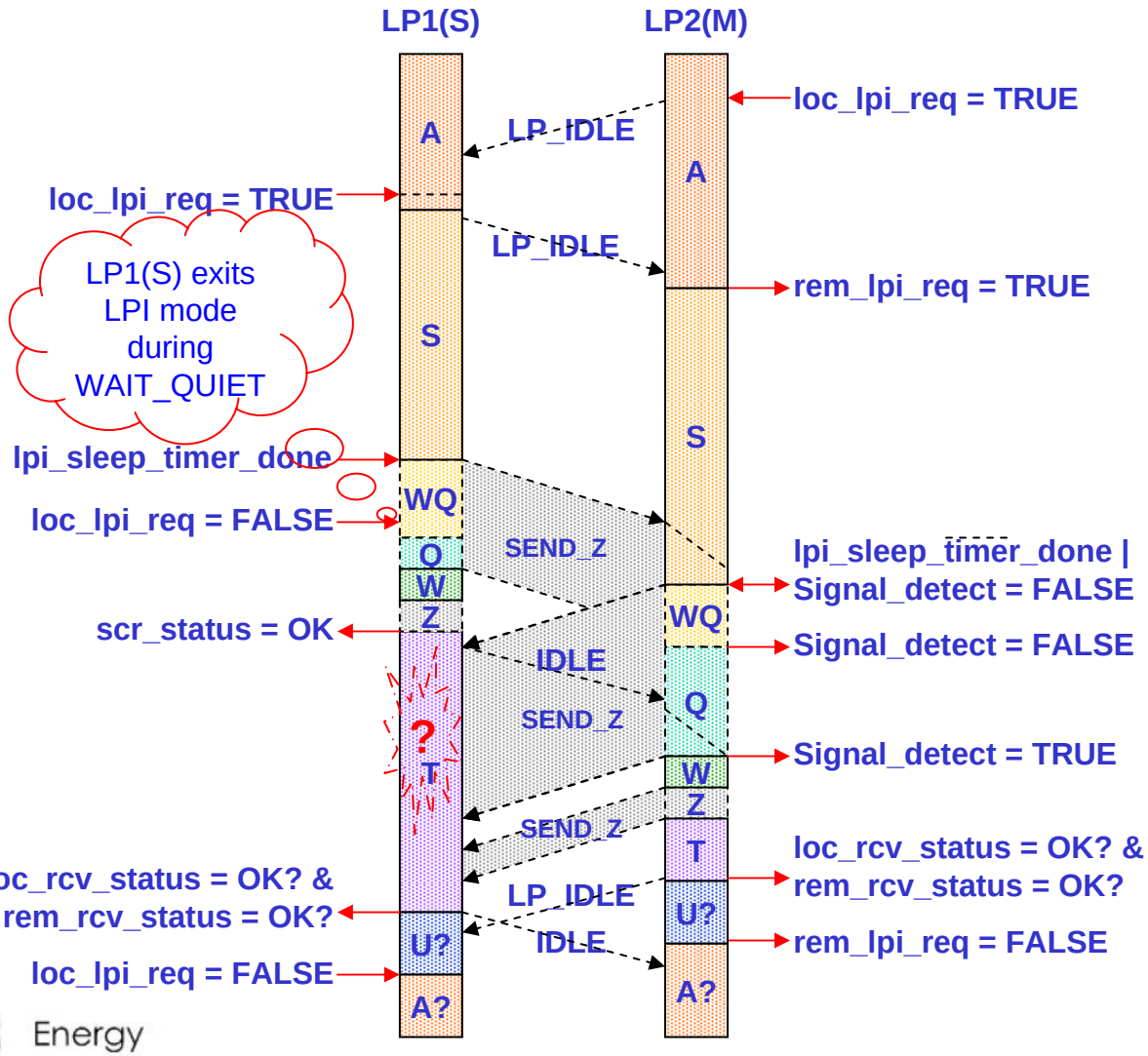
Case 2: LP1 returns to Active, LP2 goes through all transition states



1. LP2's ZERO arrives right after LP2 decides to exit LPI state
2. LP1 returns to ACTIVE state
3. LP2 in QUIET state and wait until signal_detection=TRUE, then go through all training states

➔ **LP1 will receive ZERO from LP2 for as long as one round trip delay (~1.2us) plus signal_detect time (~1us) causing timing loop failure**

Case 3: LP1(S) and LP2(M) both go through all states but out of sync



Legends

- A — Active
- S — Sleep
- WQ — Wait_Quiet
- Q — Quiet
- W — Wake
- Z — Wake_Silent
- T — Wake_Training
- U — Update
- WA — Wait_Active

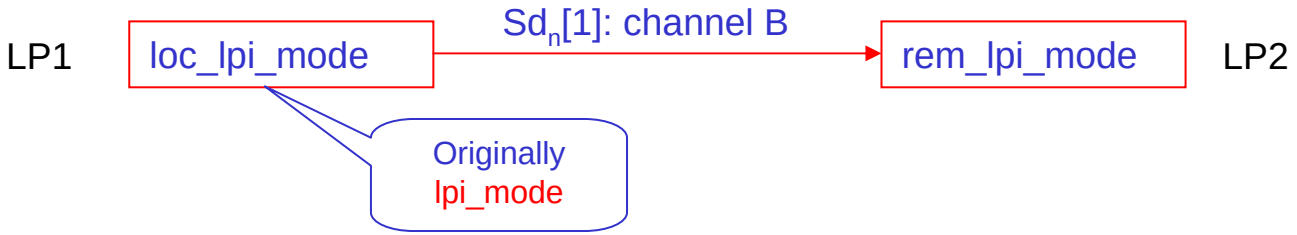
1. LP1's loc_lpi_reg = FALSE arrives before LP2 exits SLEEP state
2. LP1 goes through all training states quickly
3. LP2 will go through all training states but out of sync with LP1

➔ **LP1(S) will receive ZERO from LP2 for as long as 2.2us. The training procedure may fail.**

Comment 5 ➔ Solutions

- Encode lpi_mode signal in channel B output
- Force state SLEEP to exit to ACTIVE state if the remote party is deactivating LPI request (rem_lpi_reg = FALSE) in ACTIVE state. (case 1)
- Add a new state WAIT_ACTIVE between UPDATE and ACTIVE to avoid unwanted SEND_Z output by improved handshaking protocol . (case 2)
- Add a new watchdog timer lpi_waitact_timer (>3.6us, suggest 5us~6us)
- Add a new timer lpi_quietmin_timer (≈signal_detect time=1us) to guarantee a minimum SEND_Z time during acceleration of state transition due to loc_lpi_reg=FALSE
- Forbid exiting from intermediate state WAIT_QUIET when local LPI request is de-asserted. (case 3)

Comment 5 ➔ Solution: Encode LPI mode signal in SEND_N



$Sd_n[1]$ = { $Sc_n[1]$ ^$TXD_n[1]$, if(tx_enable_{n-2}=1); $Sc_n[1]$^ $cext\_err_n$,else if

$cext\_err_n$ = { tx_error_n if(tx_enable_n=0 *$TXD_n[7:0]$)≠0x0F *$TXD_n[7:0]$)≠0x01); 1 , if(loc_lpi_mode=TRUE *tx_mode ≠SEND_Z); 0, else

Comment 5 ➔ Solution: Modify the State machine

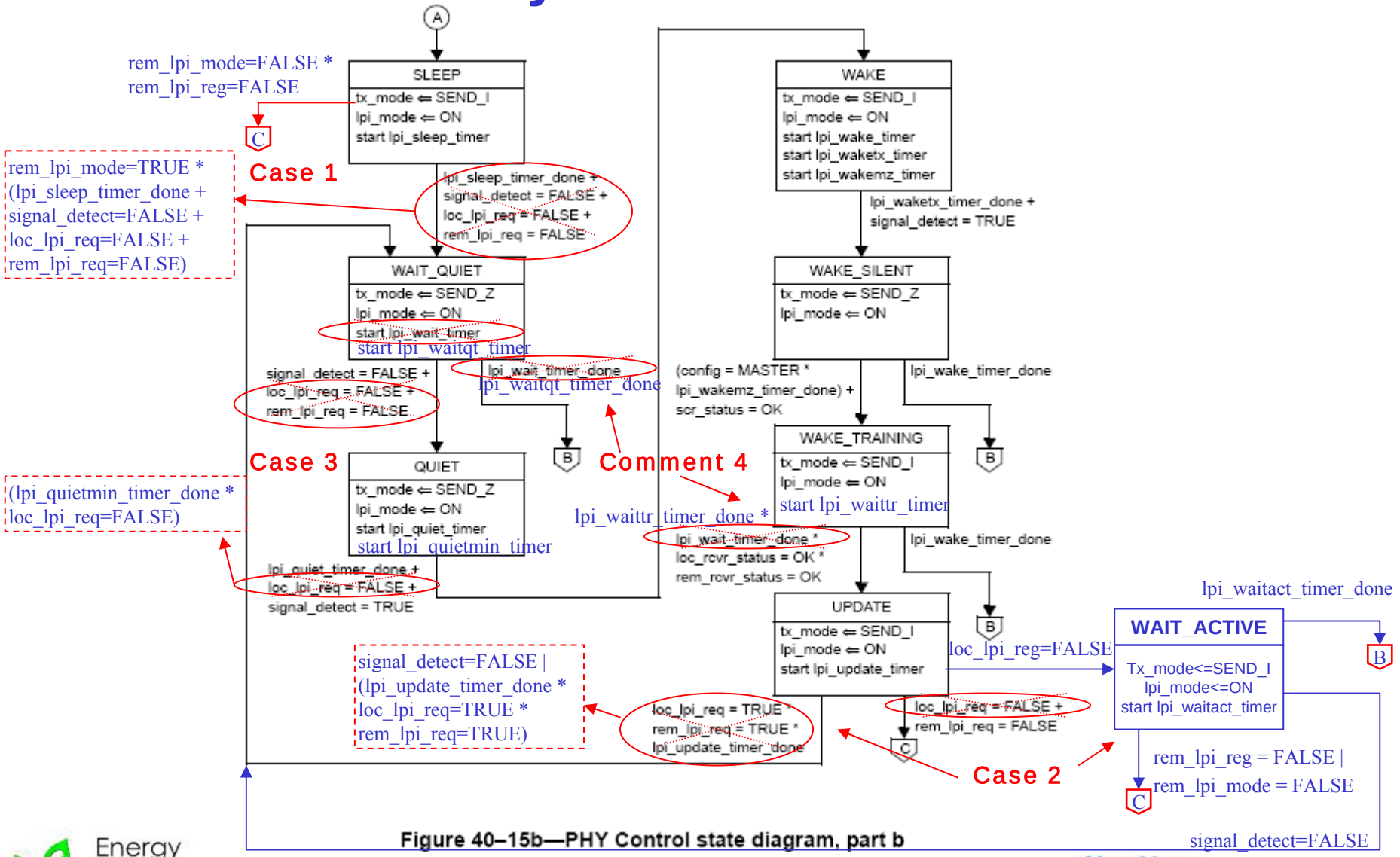
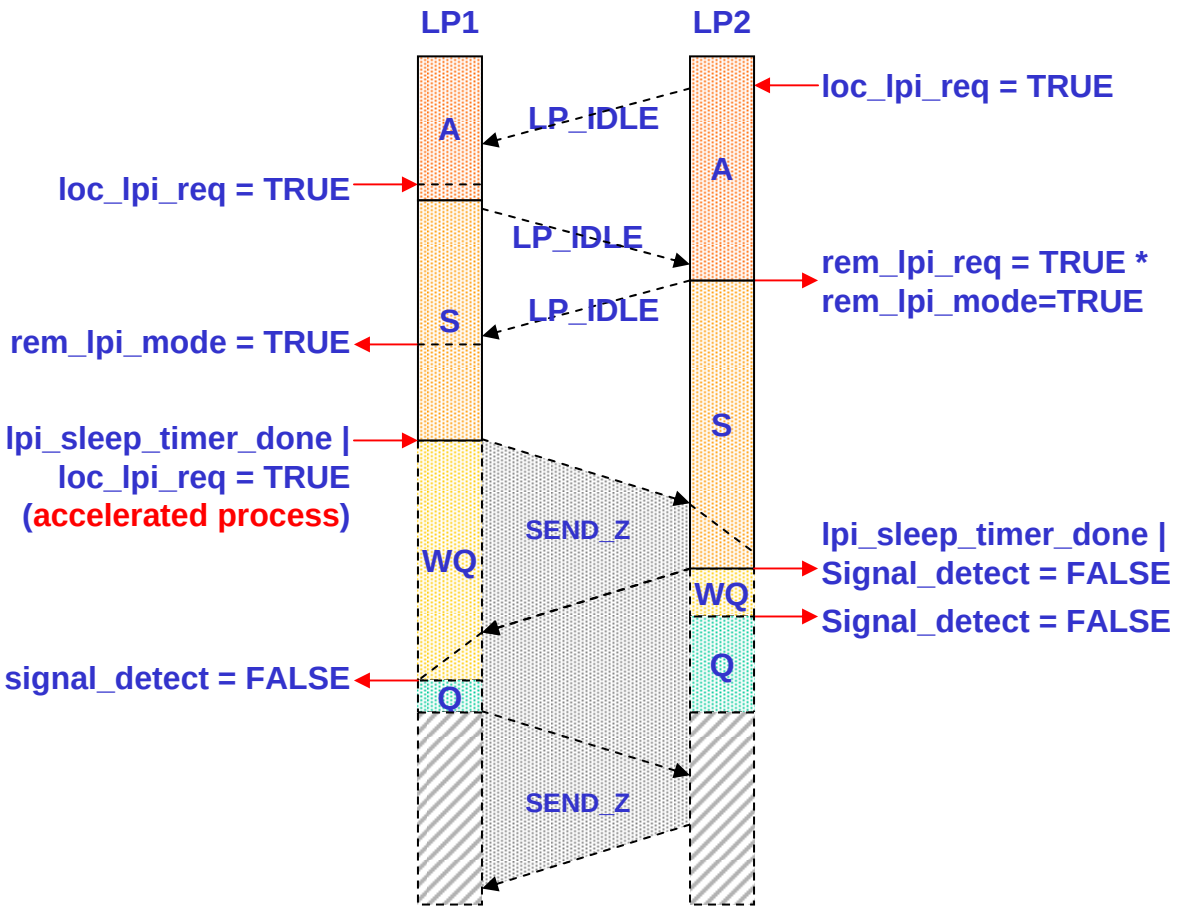
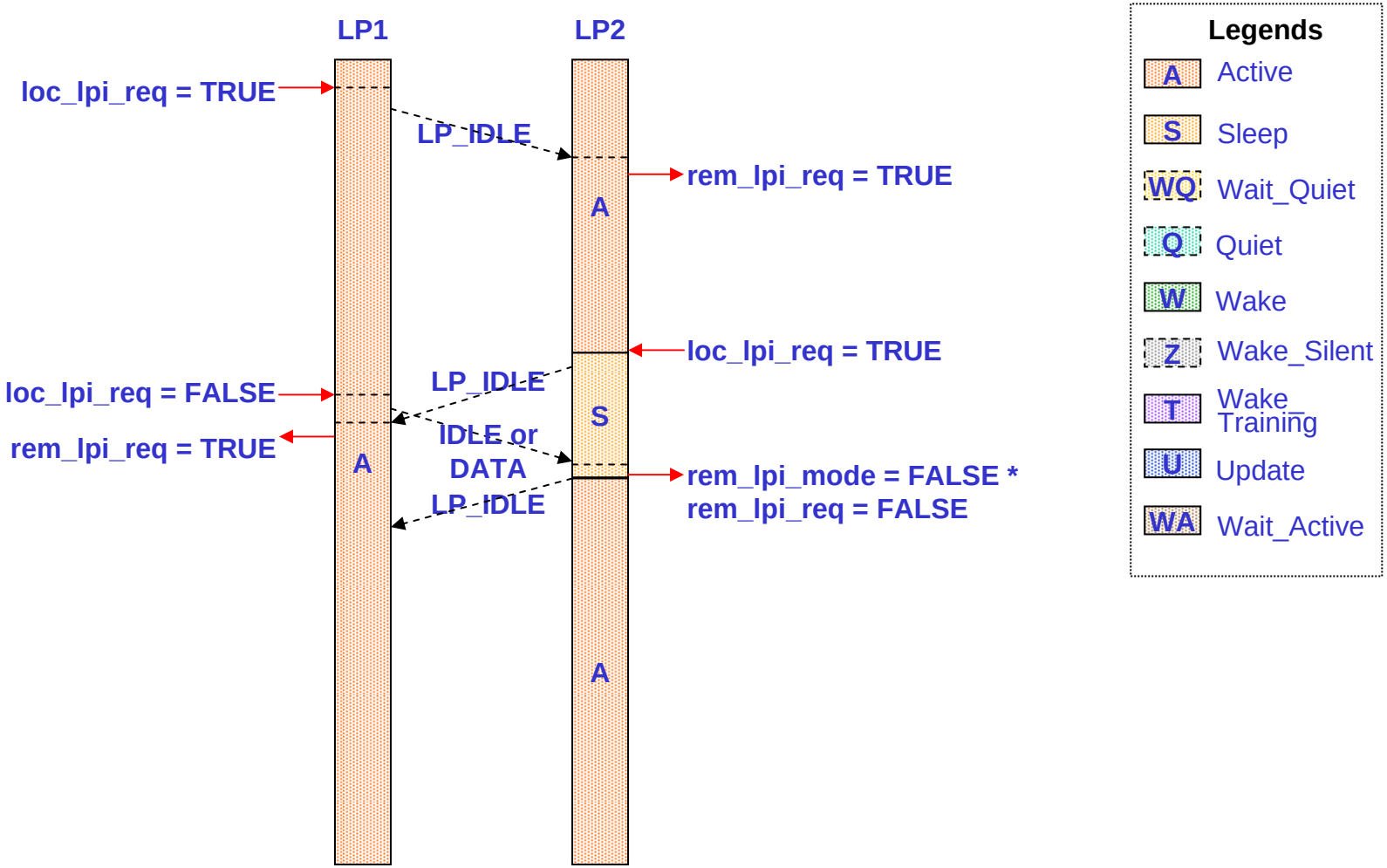


Figure 40–15b—PHY Control state diagram, part b

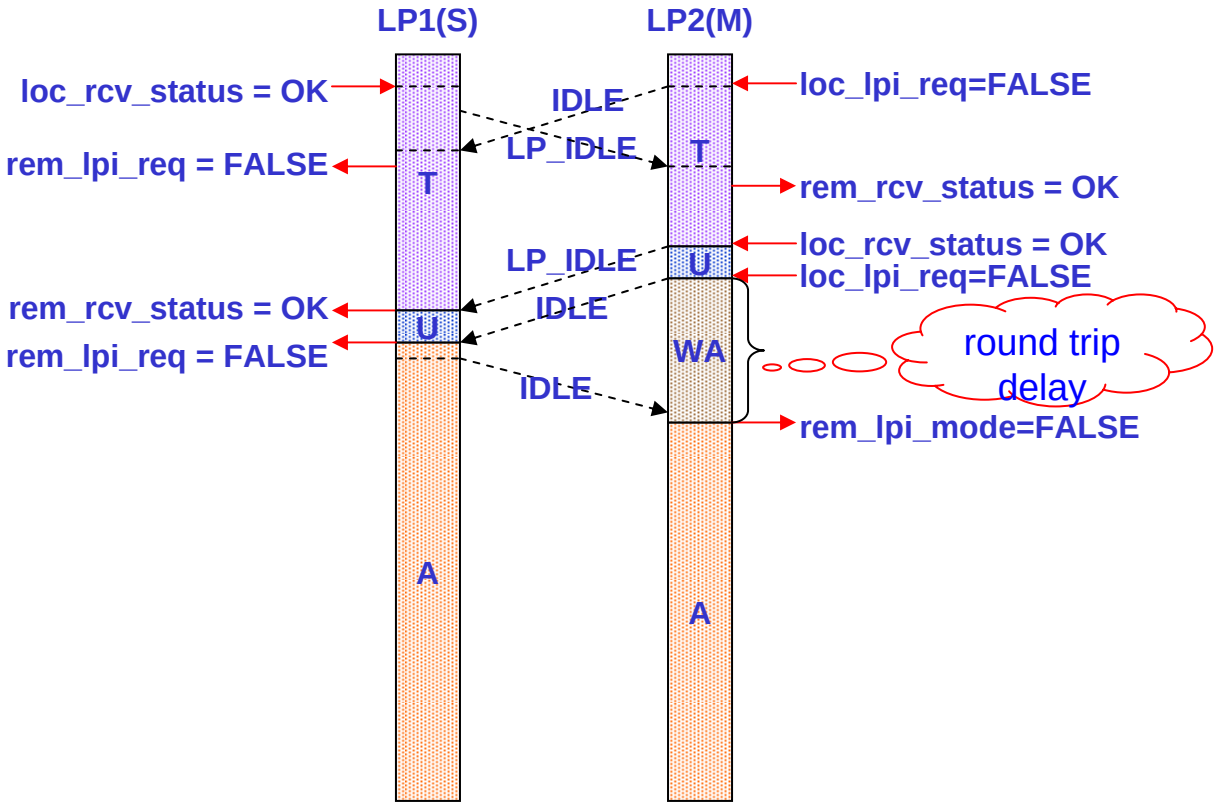
Case 1 revised: scenario 1➔ normal operation



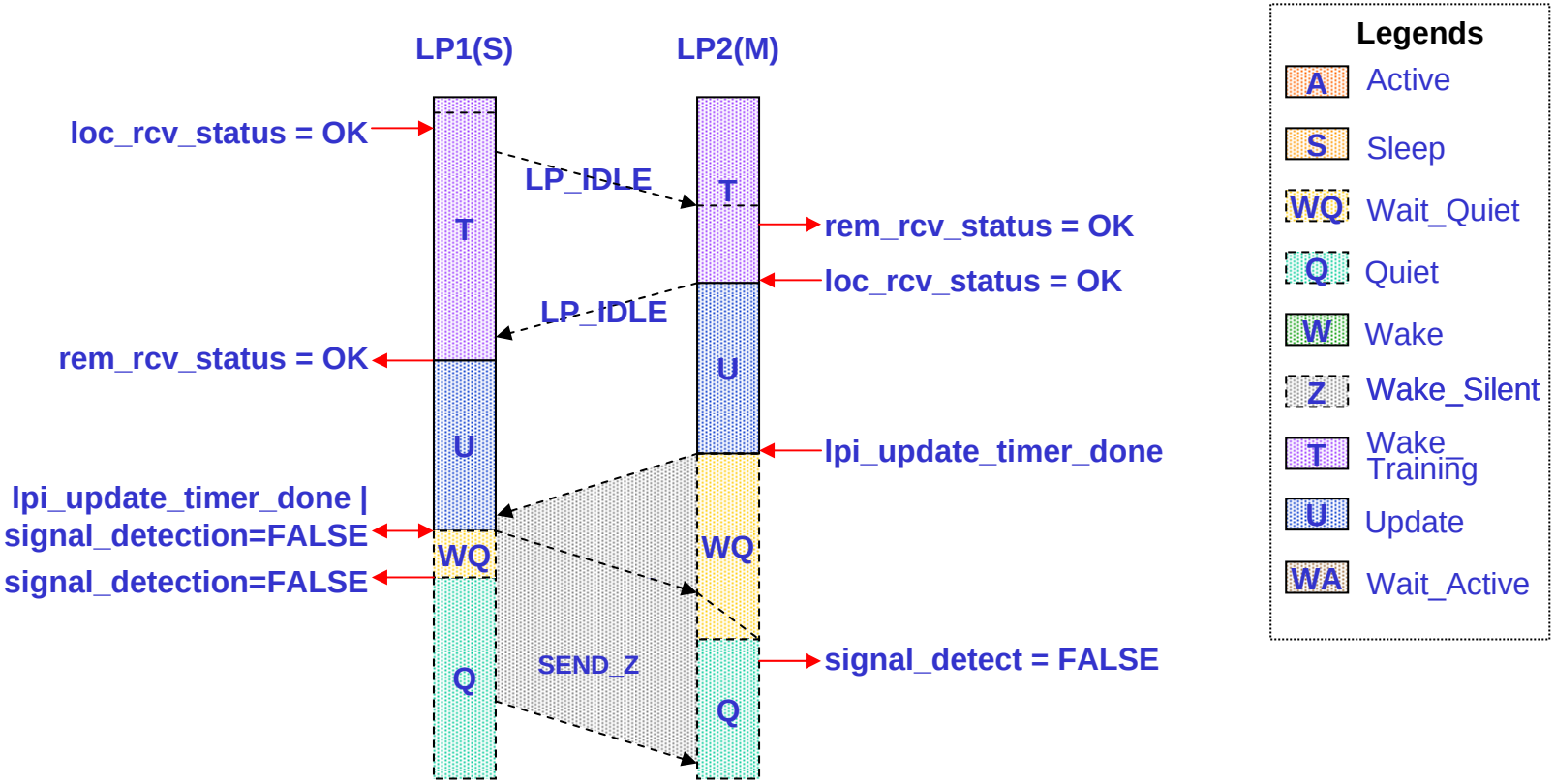
Case 1 revised: scenario 2➔ SLEEP exits to ACTIVE



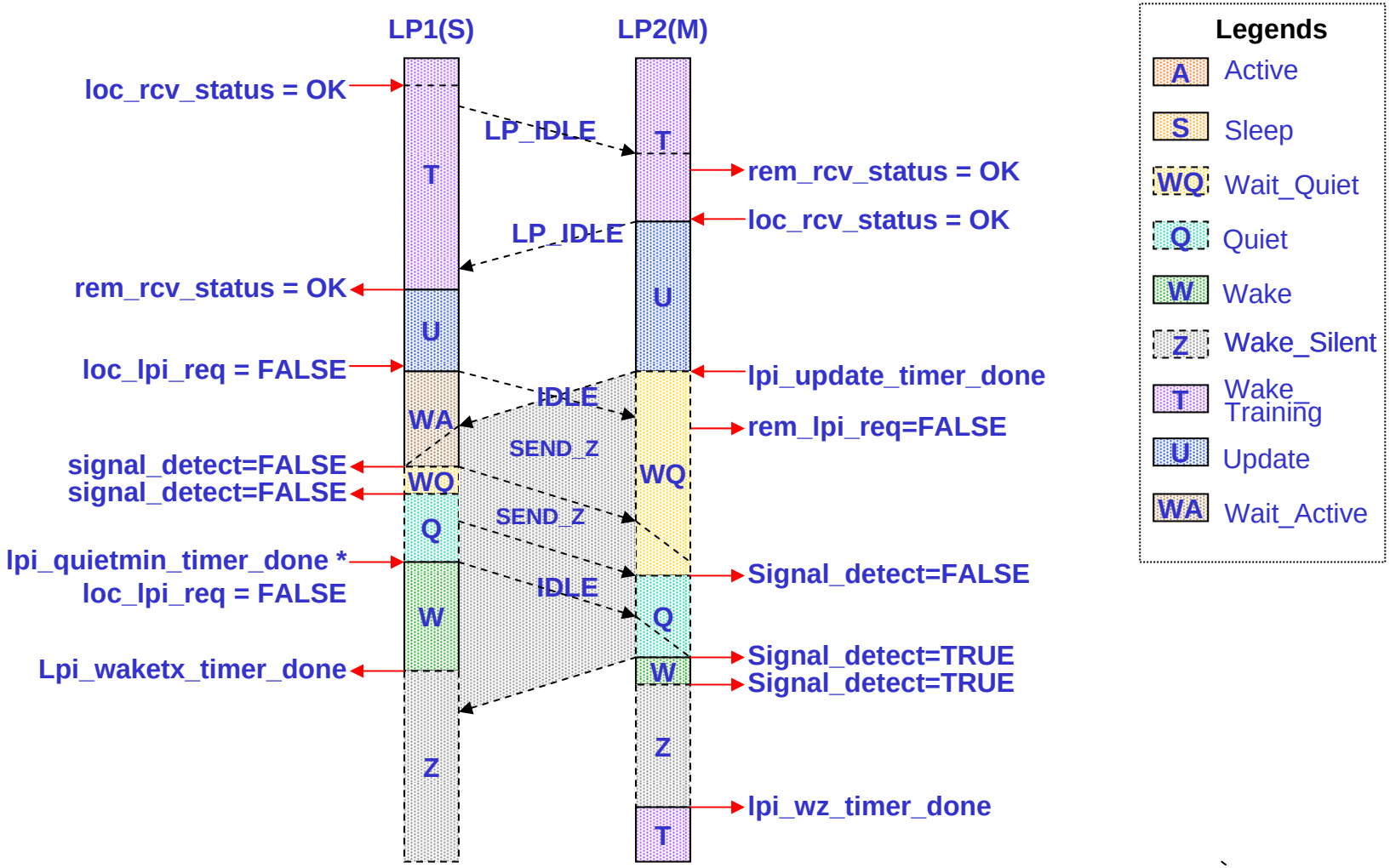
Case 2 revised: scenario 1➔Normal Wakeup



Case 2: revised: scenario 2➔Normal Refresh



Case 2 revised: scenario 3➔LP1 exits LPI; LP2 done UPDATE



Case 3: revised ➔WQ extended