Energy Efficient Ethernet 10GBASE-T LPI State Machines

Rick Tidstrom, Broadcom

IEEE P802.3az Task Force 10G ad hoc

August 6, 2008

Overview

- Propose changes to existing PCS 64B/65B Transmit and PCS 64B/65B Receive state machines.
- Add new PCS LPI Transmit State Machine.
- Add new PMA LPI Receive State Machine.
- Additions to existing state machines and new state machines will be denoted by red.
- LPI communication between the Transmitter and the Receiver has not been defined.

State Machine Assumptions

- All assumptions are a starting point, and open for discussion.

State Machine Assumptions

- State Machines support either Staggered Signaling for Quiet and Refresh, as proposed in HosseinSederat_RefreshAlertQuietSpecifications.ppt or Out of Phase Refresh as proposed on pages 8-10 in taich_02_0508.pdf.
- Details within this presentation are geared towards Staggered Signaling for Quiet and Refresh.
- From a control perspective, the Out of Phase Refresh solution is less complex.

State Machine Assumptions

- Ts, Tq, Tr, Ta, Tl, and Tw will be integer values of LDPC frames.
  - Keeps LPI controls simpler.
  - Tl = Quiet Alignment Time
  - Current LDPC frame is completed with Normal Idles when /LPI/ is detected by PCS transmitter.
  - Normal Idles during Wake need to start on LDPC frame boundary.
  - Tw could be specified at 1 64/65 block. However, change time unit from BT to T, where T = 1/800 MHz = 1.25 ns (line rate).

State Machine Assumptions

- PCS Transmit Low-Power Idle Wake-up Options:
  1. Strongly Defined Wake-up
     - Transmit PCS receives /LPI/ to stay in Low Power Idle. It receives Normal Idle to wake-up and transmit normal idle frames. Any other block type that the transmit PCS receives will cause it to wake-up and transmit wake-up error frames.
  2. Ignore all but Normal Idle Wake-up
     - Transmit PCS receives a Normal Idle to wake-up. Any other block type that is received will keep the PCS in Low Power Idle.
  3. Don't Care Wake-Up
     - Transmit PCS receives /LPI/ to stay in Low Power Idle. Any other block type that is received will cause the PCS to wake-up and transmit normal idle frames.
- The LPI state machines assume Option 1: Strongly defined wake-up.

State Machine Assumptions

- There are 3 conditions that determine the Quiet Time in ALIGN_QUIET state:

  1. 1st PHY to enter LPI uses Tq.
  2. 2nd PHY to enter LPI - align refresh with PHY already in LPI by adjusting Quiet time.
  3. Both PHYs enter LPI simultaneously:
     a) Master will enter LPI as first PHY on Channel A.
     b) Slave will enter LPI as first PHY on Channel C.

Open Issues

- When both PHYS are transmitting in Low Power Idle, the alert windows are shown as being perfectly aligned. Tolerances for the maximum skew between the alert windows needs to be defined.

Open Issues

- Active Channel behavior needs to be defined for the condition of the 2nd PHY entering LPI while the 1st PHY is transmitting Alert.
  - Propagation delay between the PCS Transmitter and the PMA receiver is Ttx + Tp (Transmitter delay + cable delay) and is approximately 2 LDPC frames (Ttx + 570 ns).
  - If Alert is four frames long, and is being transmitted by the 1st Transmitter on Channel B:
    - 1st Transmitter may be sending Alert, while Receiver is seeing Quiet. The 2nd Transmitter would transmit on Channel D, while the Receiver is expecting on Channel C.
    - 1st Transmitter may be sending Alert, while Receiver is seeing Alert. The 2nd Transmitter would transmit on Channel C, and the Receiver would expect on Channel C (When 2nd PHY is seeing Alert, it becomes 1st PHY).
    - 1st Transmitter may be sending Wake, while Receiver is seeing Alert. The 2nd Transmitter would transmit on Channel C, and the Receiver would expect on Channel C (when 2nd PHY is seeing Alert, it becomes 1st PHY).

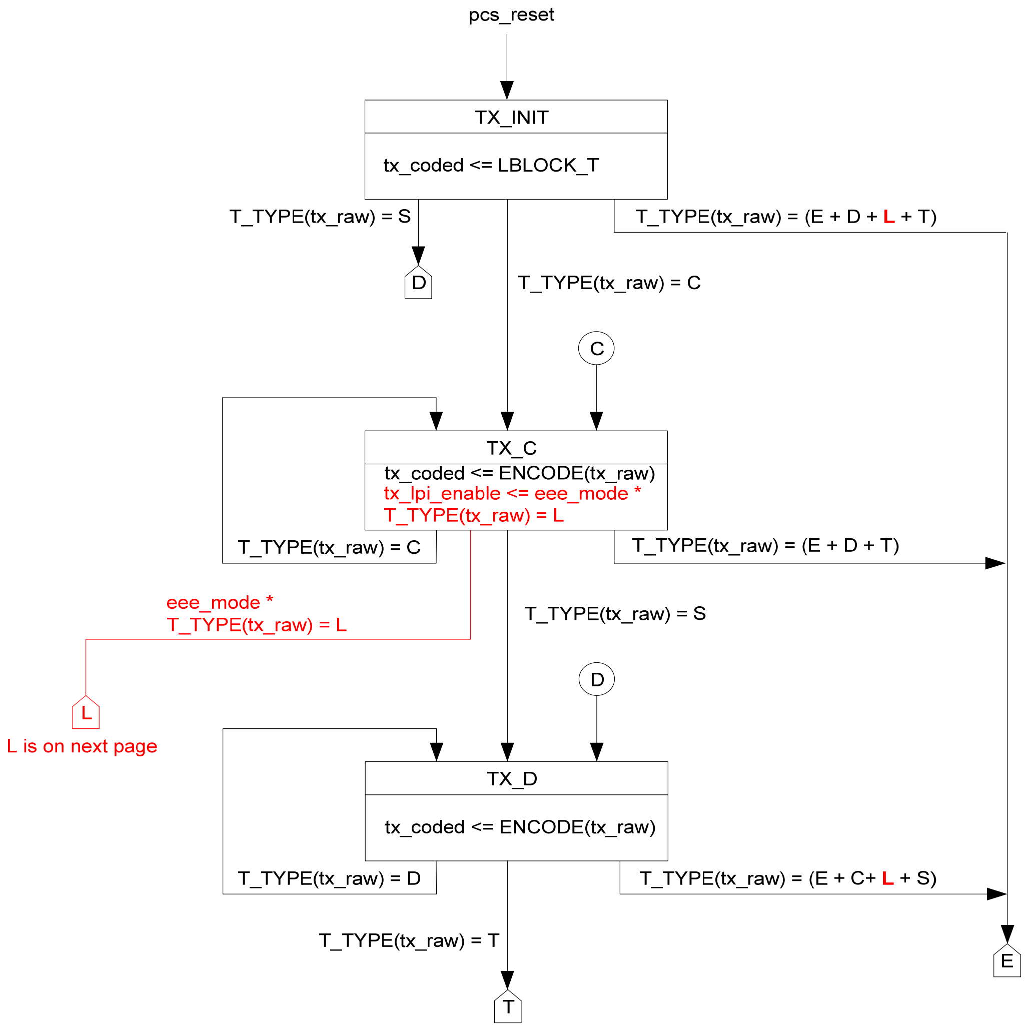
Transmit State Machine Assumptions

- PCS 64B/65B Transmit State Machine
  - TX_INIT state can not go straight to TX_L state (Low Power Idle State)
  - tx_lpi_enable goes active in TX_C state due to transition condition of eee_mode * T_TYPE(tx_raw) = L. This is not necessary if:
    - T_TYPE(tx_raw) is mapped into Normal Idle in 64B/65B state machine.
    - PCS Transmitter starts sending /LPI/ blocks immediately (instead of completing current LDPC frame with Normal Idles)
  - Wake-up from TX_L is strongly defined (must see Normal Idle to wake-up).

Transmit State Machine Assumptions

- PCS LPI Transmit State Machine
  - New state machine to support Low Power Idle.
  - It was split from 64B/65B state machine based for the following reasons:
    - 64B/65B state machine is driven by values received across the XGMII.
    - LPI state machine is LPI timer driven.
  - It will be located in the PCS transmit layer:
    - Needs to respond to Sleep and Wake, which are XGMII commands.
    - LDPC timing is in the PCS layer.
  - ALIGN_QUIET state does not handle Wake-up.
  - When 2nd PHY to enter LPI enters while the 1st PHY is transmitting Refresh, it will transmit Quiet, and add remaining Refresh time to Tq. The 2nd PHY will also anticipate the 1st PHY transitioning to a new channel.
  - When /LPI/ command is received, current LDPC frame will be completed with Normal Idles (to keep Ts an LDPC frame integer value).

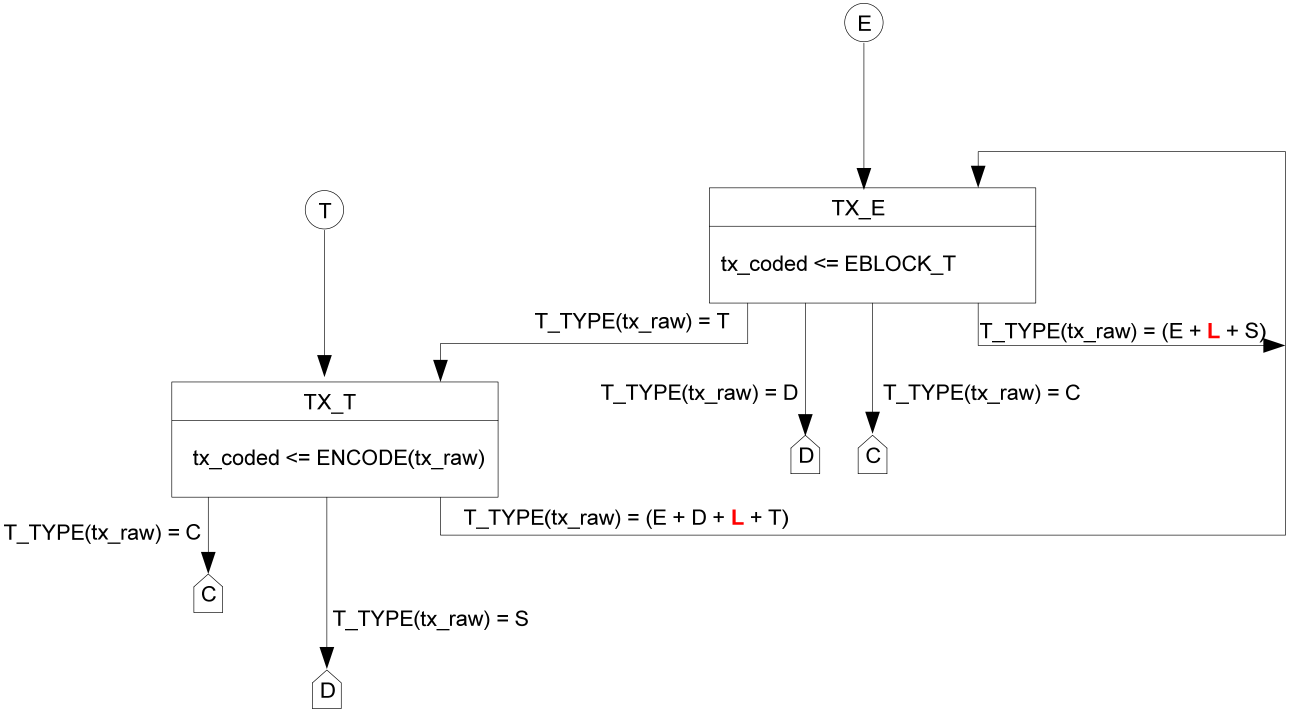
PCS 64B/65B Transmit FSM (1 of 3)



Energy Efficient Ethernet

BROADCOM.
Connecting everything®

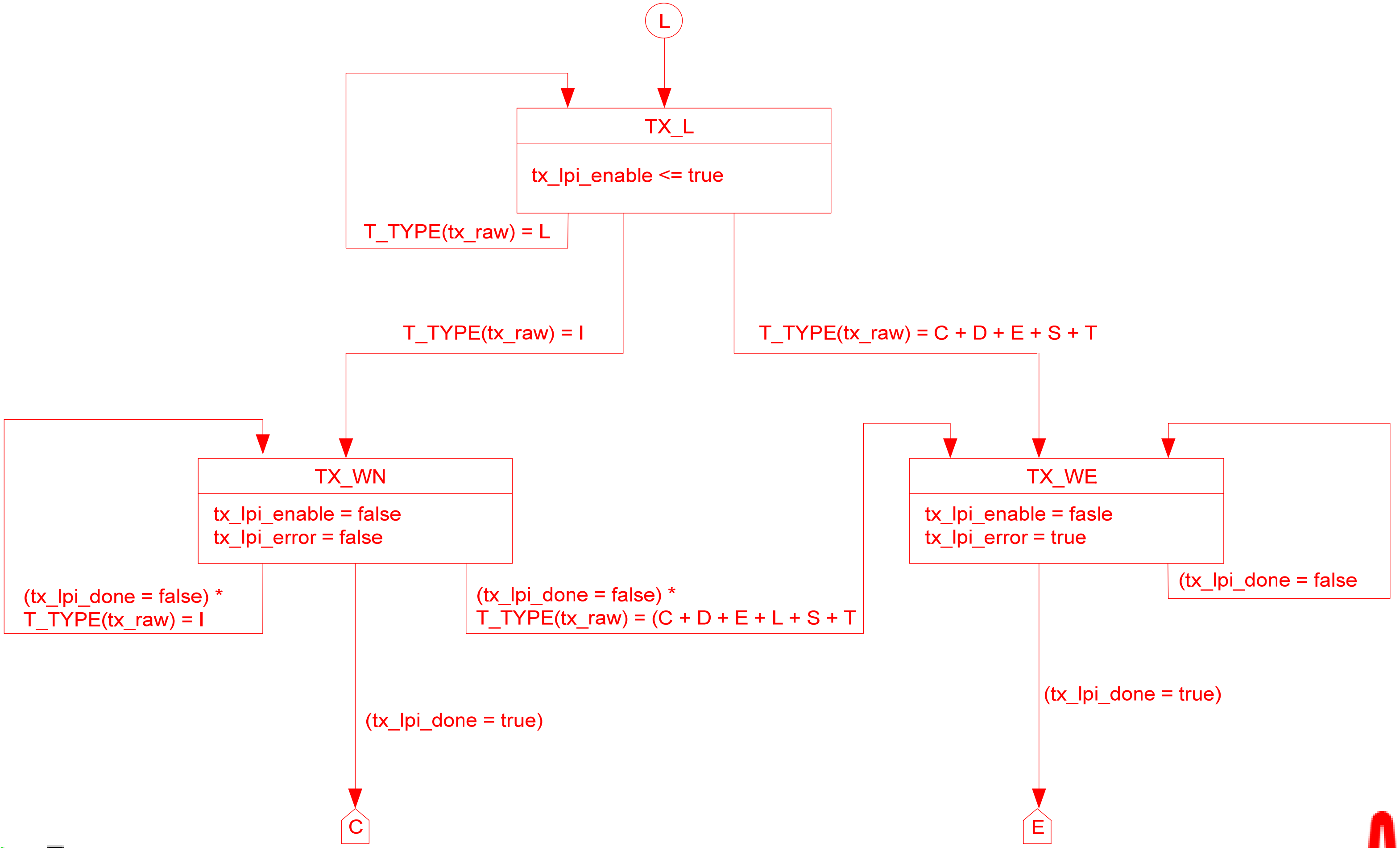
PCS 64B/65B Transmit FSM (2 of 3)



Energy Efficient Ethernet

BROADCOM. Connecting everything®

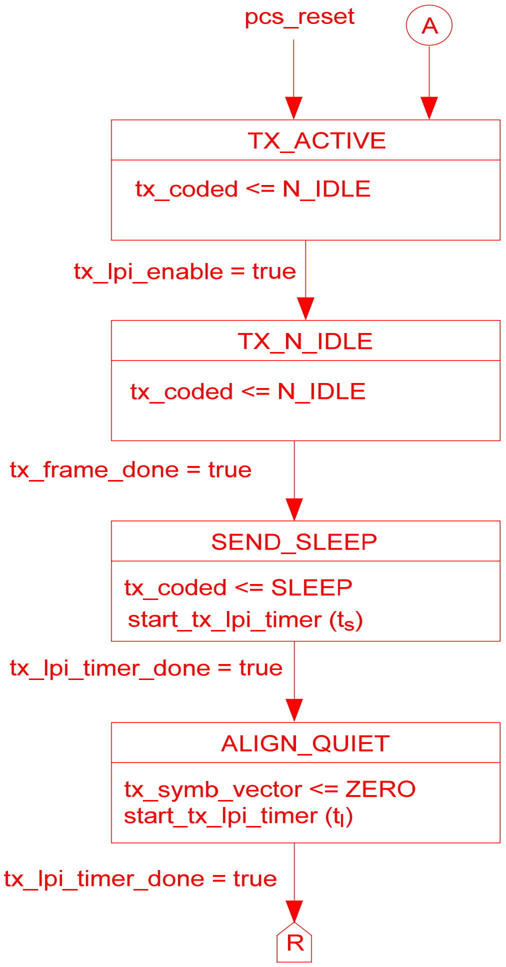
PCS 64B/65B Transmit FSM with Option 1 Wake-up (3 of 3)



Energy Efficient Ethernet

BROADCOM. Connecting everything®

PCS LPI Transmit FSM (1 of 2)



pcs_reset

A

TX_ACTIVE

tx_coded <= N_IDLE

tx_lpi_enable = true

TX_N_IDLE

tx_coded <= N_IDLE

tx_frame_done = true

SEND_SLEEP

tx_coded <= SLEEP
start_tx_lpi_timer (t_S)

tx_lpi_timer_done = true

ALIGN_QUIET

tx_symb_vector <= ZERO
start_tx_lpi_timer (t_l)

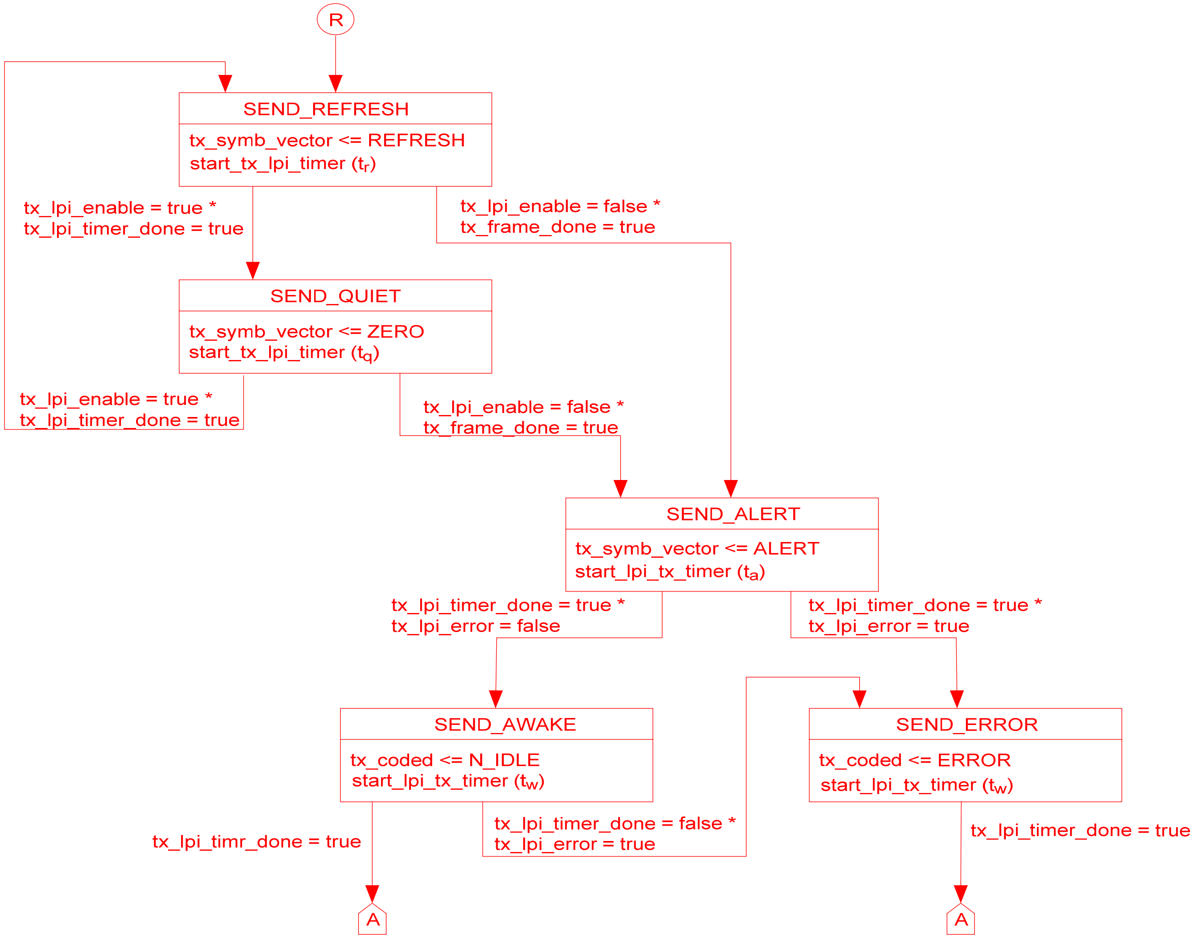
tx_lpi_timer_done = true

R

Energy
Efficient
Ethernet

BROADCOM.
Connecting everything®

PCS LPI Transmit FSM (2 of 2)



R

SEND_REFRESH

tx_symb_vector <= REFRESH
start_tx_lpi_timer (t_r)

tx_lpi_enable = true *
tx_lpi_timer_done = true

tx_lpi_enable = false *
tx_frame_done = true

SEND_QUIET

tx_symb_vector <= ZERO
start_tx_lpi_timer (t_q)

tx_lpi_enable = true *
tx_lpi_timer_done = true

tx_lpi_enable = false *
tx_frame_done = true

SEND_ALERT

tx_symb_vector <= ALERT
start_lpi_tx_timer (t_a)

tx_lpi_timer_done = true *
tx_lpi_error = false

tx_lpi_timer_done = true *
tx_lpi_error = true

SEND_AWAKE

tx_coded <= N_IDLE
start_lpi_tx_timer (t_w)

tx_lpi_timr_done = true

tx_lpi_timer_done = false *
tx_lpi_error = true

SEND_ERROR

tx_coded <= ERROR
start_lpi_tx_timer (t_w)

tx_lpi_timer_done = true

A

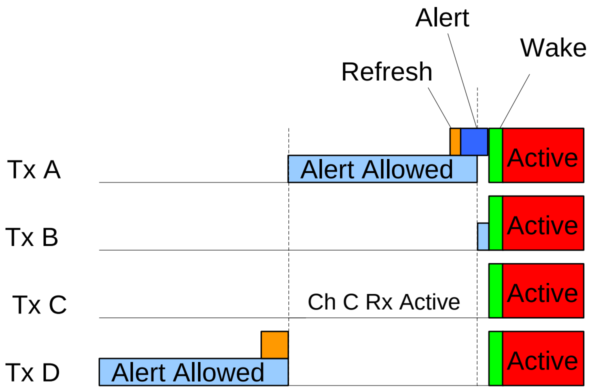
A

Receive State Machine Assumptions

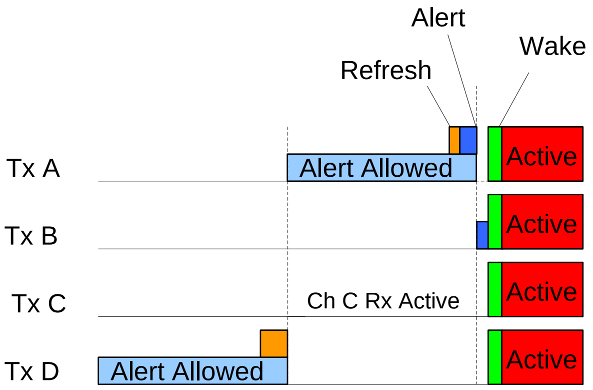
- PCS LPI Receive State Machine
  - New state machine to support Low Power Idle.
  - It was placed in PMA layer, because PCS layer will not receive data during LPI.
  - It was split from 64B/65B state machine for the following reasons:
    - 64B/65B state machine is driven by values received across the XGMII.
    - 64B/65B state machine deals with LDPC encoded blocks.
    - LPI state machine is LPI timer driven.
    - LPI state machine deals with PAM-2 symbols.
  - Conditional transition out of ALIGN_QUIET state uses energy detect instead of rx_lpi_timer_done because alignment time value is only approximate.

Receive State Machine Assumptions

- When Alert occurs during Refresh, it will finish on the same channel that it started on (Option 1), instead of transitioning to the next channel when the Alert Allowed window expires (Option 2).

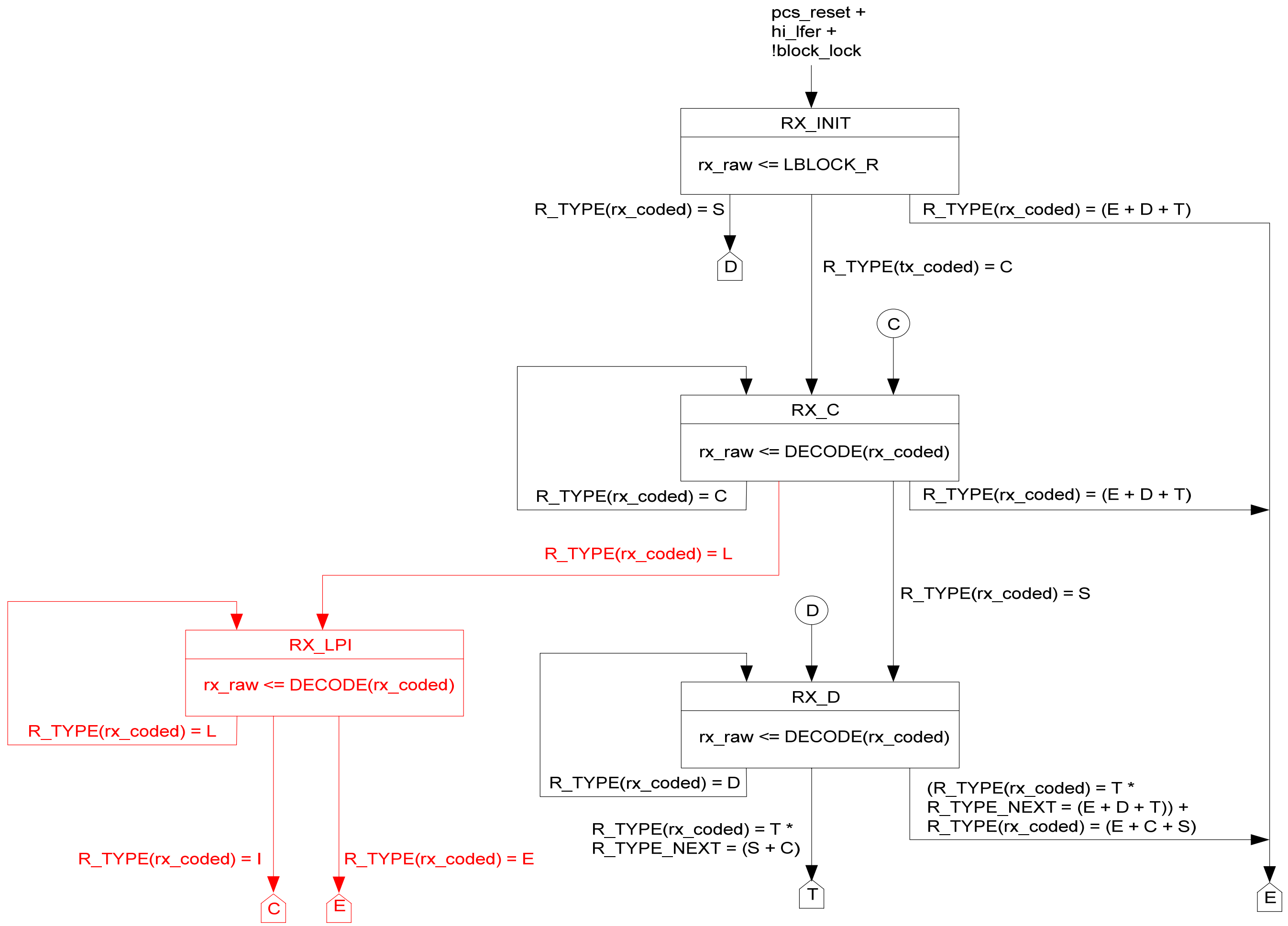


Option 1: Finish Alert on same Channel that it started



Option 2: Split Alert. Finish on next Channel.

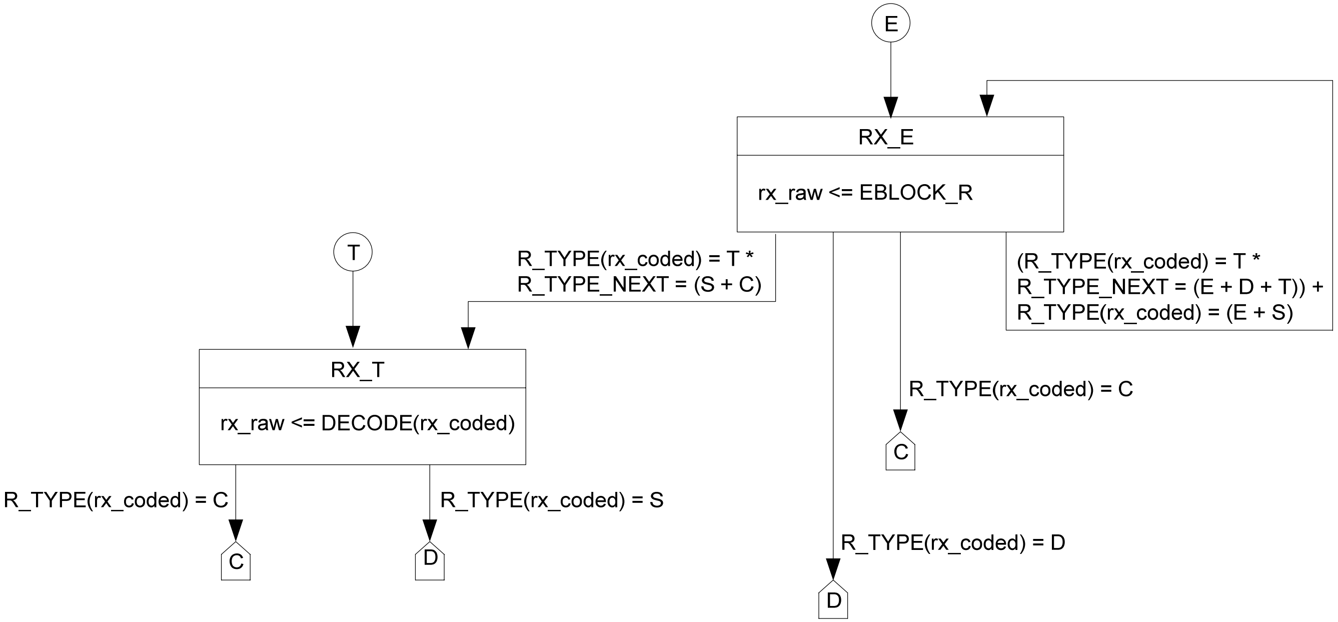
PCS 64B/65B Receive FSM (1 of 2)



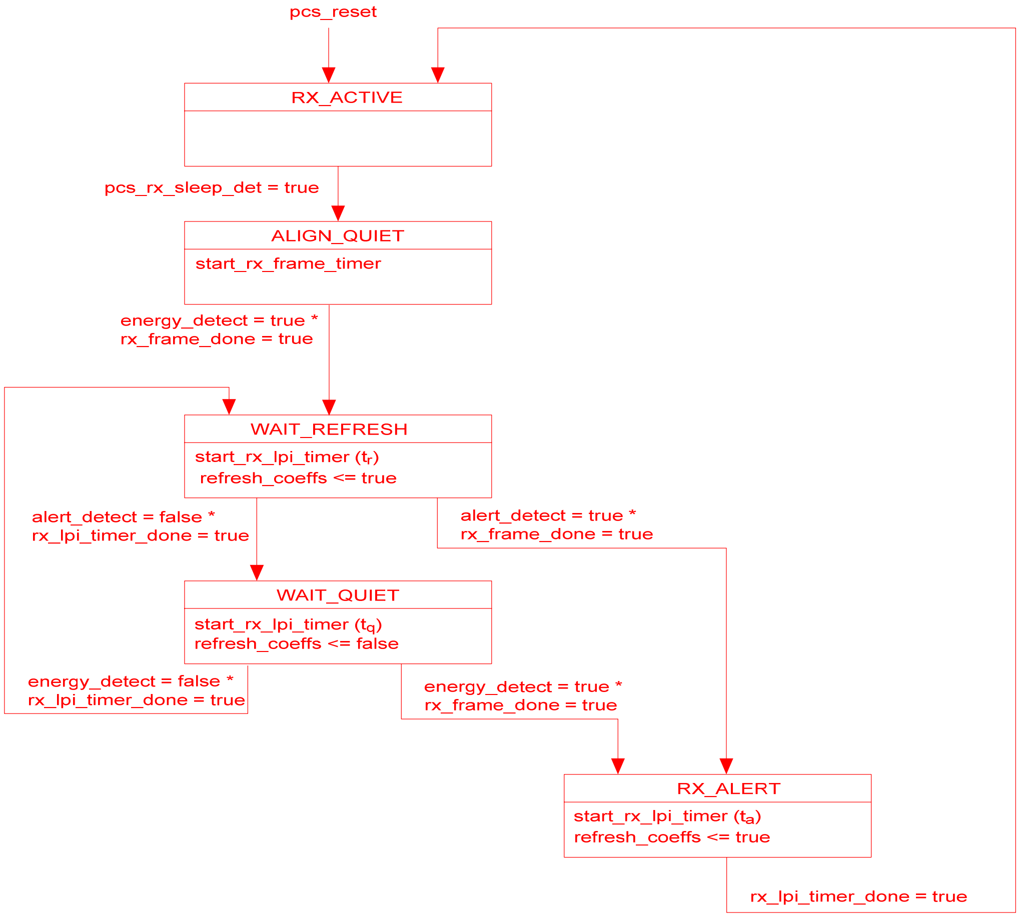
Energy Efficient Ethernet

BROADCOM. Connecting everything®

PCS 64B/65B Receive FSM (2 of 2)



PMA LPI Receive FSM (1 of 1)



pcs_reset

RX_ACTIVE

pcs_rx_sleep_det = true

ALIGN_QUIET

start_rx_frame_timer

energy_detect = true *
rx_frame_done = true

WAIT_REFRESH

start_rx_lpi_timer (t_r)
refresh_coeffs <= true

alert_detect = false *
rx_lpi_timer_done = true

alert_detect = true *
rx_frame_done = true

WAIT_QUIET

start_rx_lpi_timer (t_q)
refresh_coeffs <= false

energy_detect = false *
rx_lpi_timer_done = true

energy_detect = true *
rx_frame_done = true

RX_ALERT

start_rx_lpi_timer (t_a)
refresh_coeffs <= true

rx_lpi_timer_done = true