ILT Training FSM Update

comments #463, 45, 230, 481

Jeff Slavick (Broadcom)
Adee Ran (Cisco)

Supporters

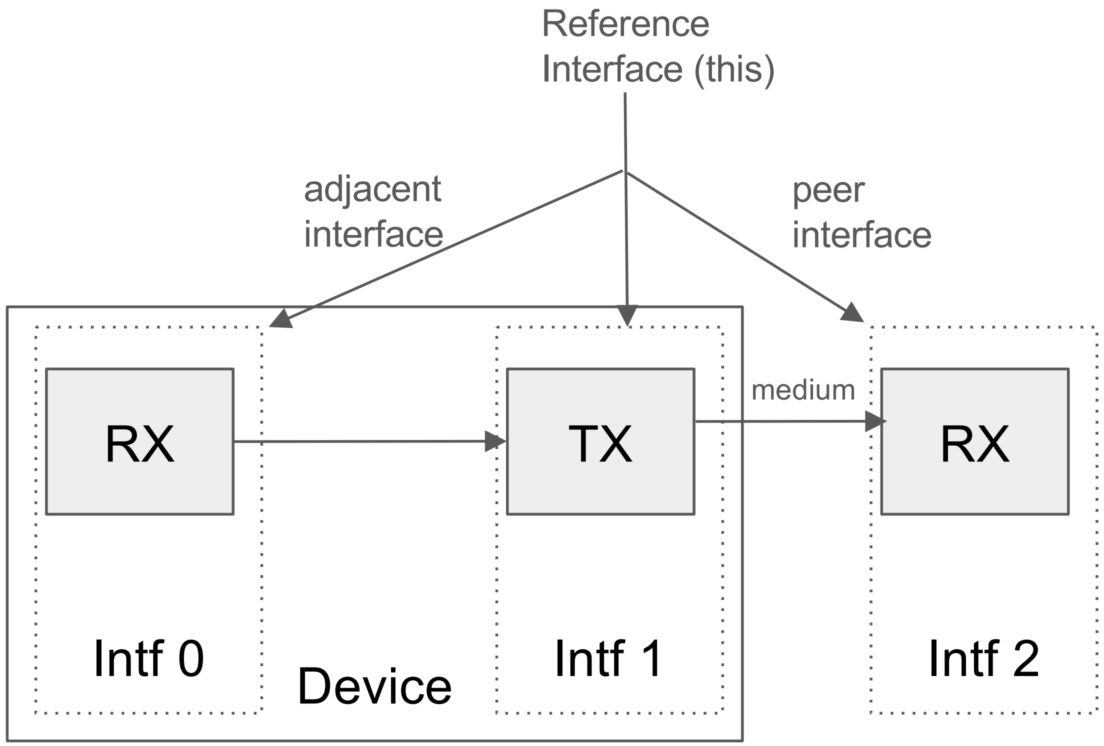
Nomenclature

Adjacent interface:

Interface whose service interface is connected (directly or indirectly) with this interface (logical). There may be multiple sub-layers between the two ILT interfaces.

Peer interface:

Interface that's connected with this interface through a medium of some type (physical)



Example of Peer and Adjacent Interfaces in a Link

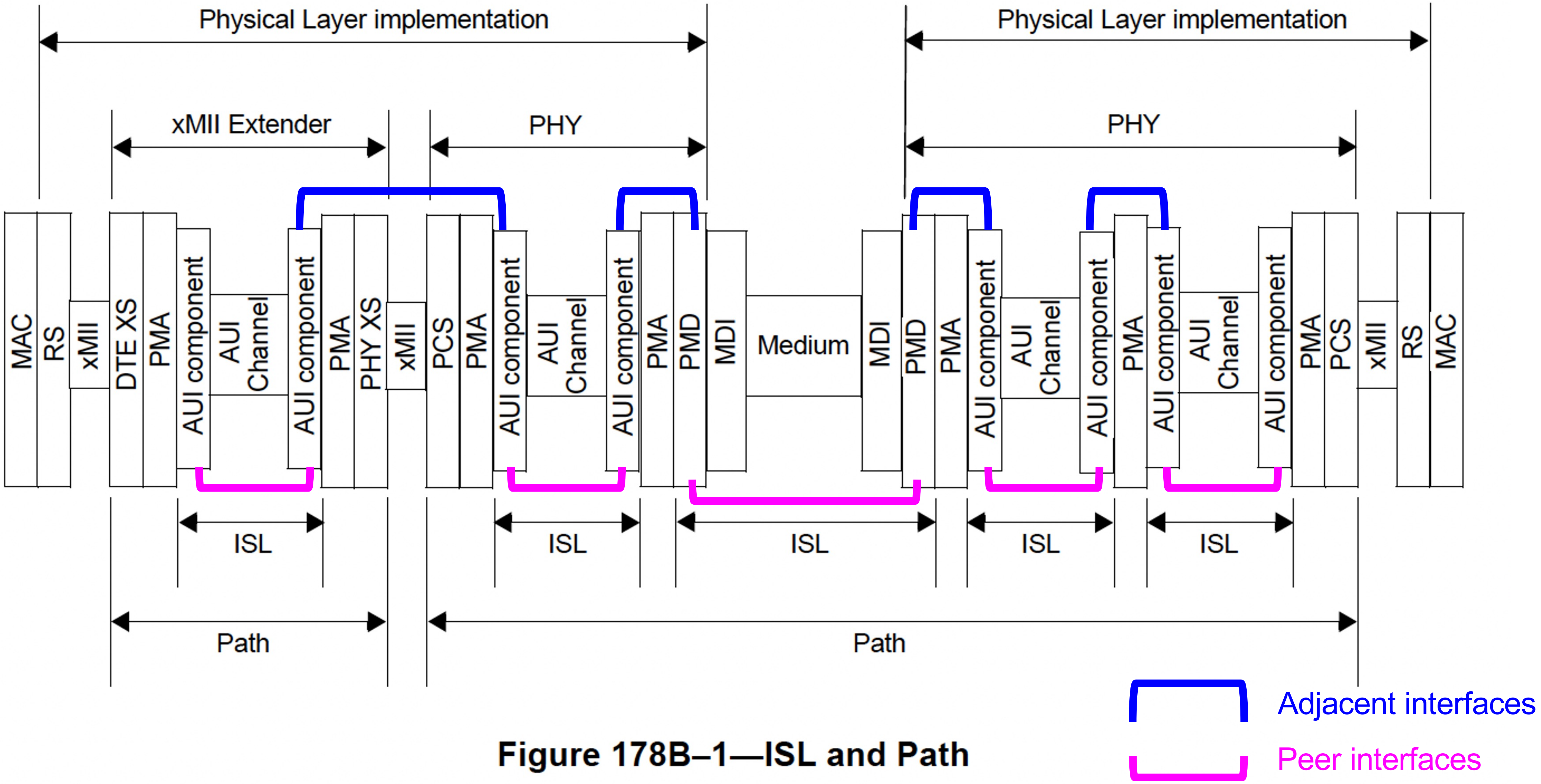
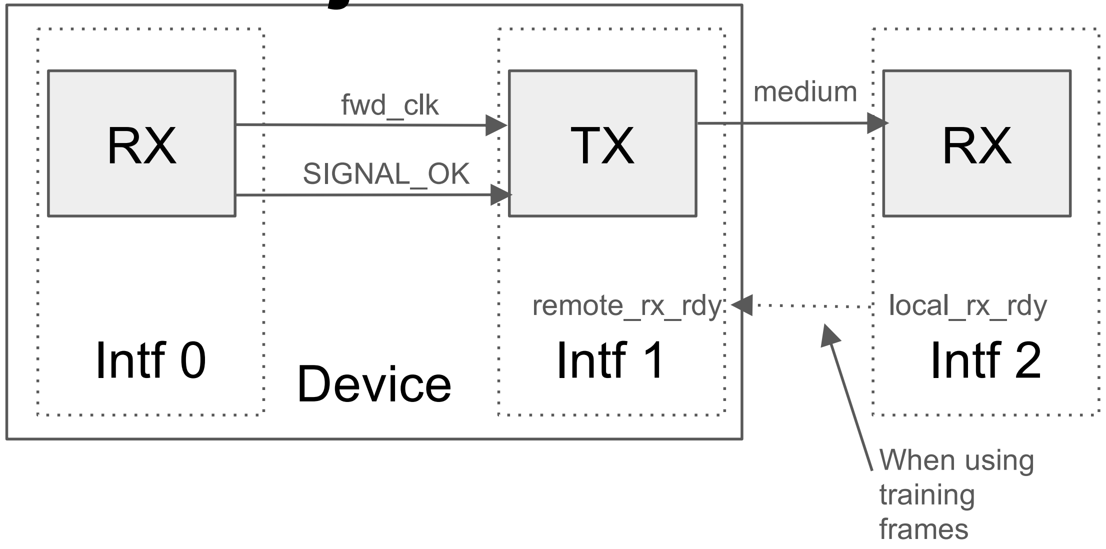


Figure 178B–1—ISL and Path

Flow of the start-up process between adjacent ISLs



When using training frames, an interface has information about the peer interface state, the adjacent interface and its own state.

This permits starting training with the peer interface using a local clock prior to the adjacent interface supplying a stable data mode clock (fwd_clk).

The swap from local clock to data mode clock occurs when adjacent interface, peer interface and local interface are fully adapted and the adjacent interface Rx is receiving data mode clock..

The Ready-To-Send (RTS) indicator is used to denote the signal is operating on data mode clock. Path start-up process cascades the RTS indicator down the path beginning from the RS. This is encoded in the Continue Training field of the training frames.

ILT FSMs

There are two state diagrams that control the behavior of the Start-up process.

Per interface RTS update state diagram (Figure 178B-9) currently managing:

- Clock Forwarding state
- Service Interface SIGNAL_OK state set by training_status variable
- RTS indicator

Per lane Training control state diagram (Figure 178B-10) currently managing:

- Tx data mode & pattern
- Transmit disable state
- Per lane training/receiver status

RTS update state diagram (Figure 178B-9)

This diagram is new for ILT compared to legacy PMD Control functions (Clause 162, 136, 93, 72).

Its addition is to support Path-Startup and should be limited to changing the Tx state to “Ready To Send” state.

This assignment of the training_status variable needs to be removed from (Figure 178B-9) as that’s information to be sent to another interface, not used by the local interface.

Operation with training (mr_training_enable=true)

How do we communicate these states?

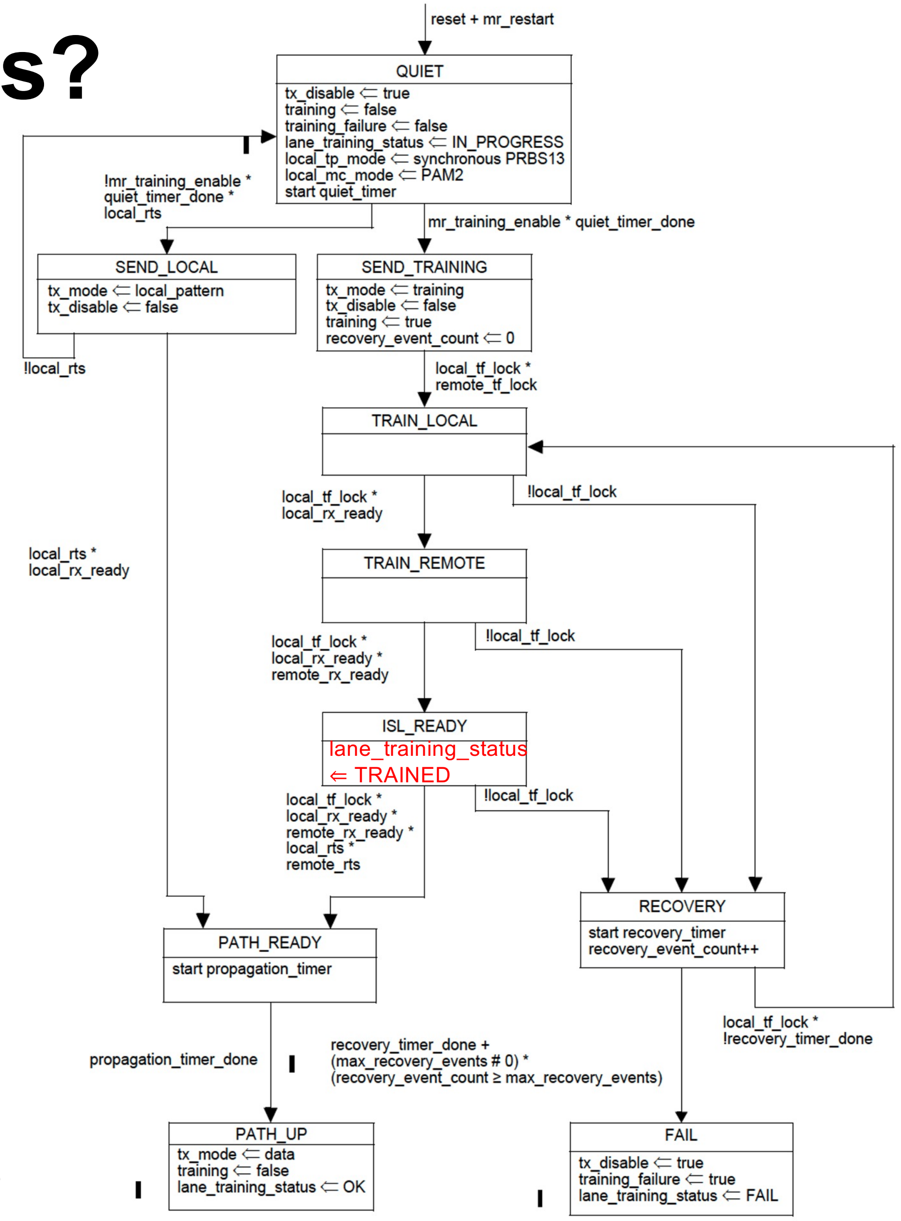
peer & local rx rdy: Existing definition of isl_ready is valid. After local and peer Rx are done adapting.

Adj Rx done adapting: Add a new lane_training_status state of TRAINED that indicates local_rx_ready and remote_rx_ready are both true. This is used in training_status state generation which is mapped to SIGNAL_OK

Adj Rx receiving data clk: Existing definition of remote_rts is valid. (true when CT=0 on all lanes) This is mapped in to SIGNAL_OK

SIGNAL_OK primitive is used to transfer the above states to the adjacent interface and is mapped from training_status variable. Define the states of training_status to be as follows:

FAIL: Any lane_training_status is FAIL
OK: All lane_training_status are OK
READY: remote_rts is true AND all lanes have lane_training_status of TRAINED or OK with at least one lane being TRAINED.
IN_PROGRESS: At least one lane has lane_training_status of IN_PROGRESS and no lanes have lane_training_status of FAIL.



reset + mr_restart
QUIET
tx_disable ⇐ true
training ⇐ false
training_failure ⇐ false
lane_training_status ⇐ IN_PROGRESS
local_tp_mode ⇐ synchronous PRBS13
local_mc_mode ⇐ PAM2
start quiet_timer
!mr_training_enable * quiet_timer_done * local_rts
mr_training_enable * quiet_timer_done
SEND_LOCAL
tx_mode ⇐ local_pattern
tx_disable ⇐ false
!local_rts
SEND_TRAINING
tx_mode ⇐ training
tx_disable ⇐ false
training ⇐ true
recovery_event_count ⇐ 0
local_tf_lock * remote_tf_lock
TRAIN_LOCAL
local_tf_lock * local_rx_ready
!local_tf_lock
local_rts * local_rx_ready
TRAIN_REMOTE
local_tf_lock * local_rx_ready * remote_rx_ready
!local_tf_lock
ISL_READY
lane_training_status ⇐ TRAINED
local_tf_lock * local_rx_ready * remote_rx_ready * local_rts * remote_rts
!local_tf_lock
RECOVERY
start recovery_timer
recovery_event_count++
PATH_READY
start propagation_timer
local_tf_lock * !recovery_timer_done
propagation_timer_done
recovery_timer_done + (max_recovery_events # 0) * (recovery_event_count ≥ max_recovery_events)
PATH_UP
tx_mode ⇐ data
training ⇐ false
lane_training_status ⇐ OK
FAIL
tx_disable ⇐ true
training_failure ⇐ true
lane_training_status ⇐ FAIL

Figure 178B–10—Training control state diagram

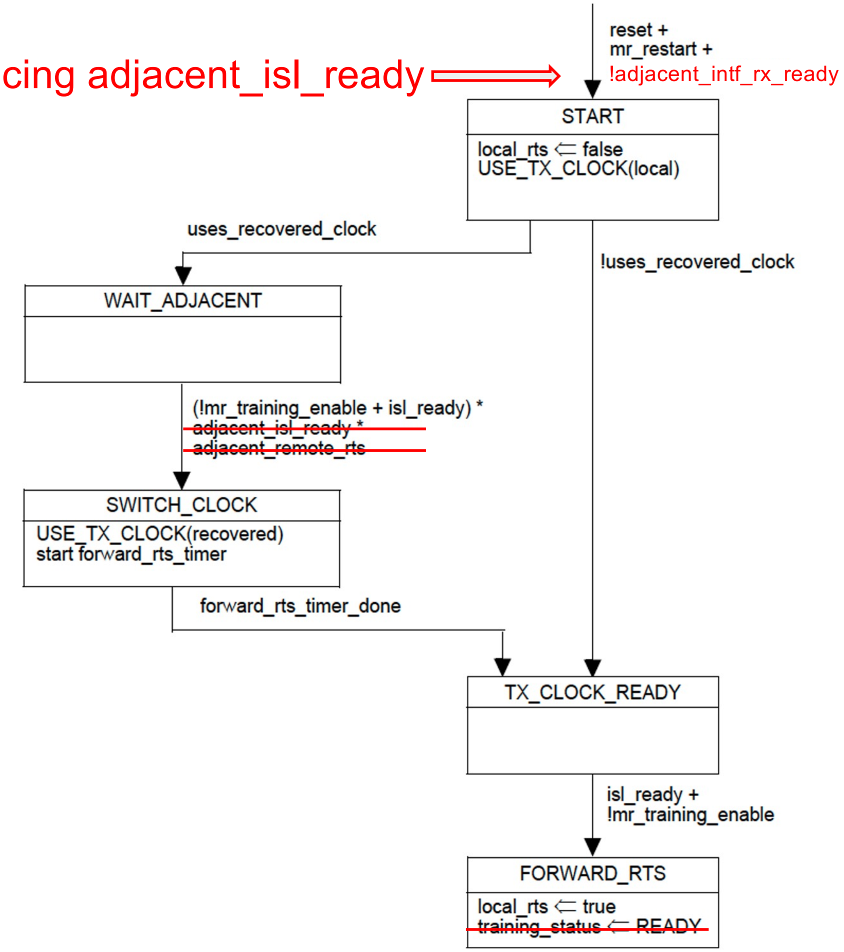
Make RTS update state diagram be just about the Tx

Remove adjacent_remote_rts and adjacent_isl_ready variables definitions and add adjacent_intf_rx_ready.

adjacent_intf_rx_ready

Boolean variable that is derived from the received SIGNAL_OK primitive on the service interface. It is set to TRUE when SIGNAL_OK is either READY or OK and FALSE otherwise.

Replacing adjacent_isl_ready ⟹



reset +
mr_restart +
!adjacent_intf_rx_ready

START

local_rts ⇐ false
USE_TX_CLOCK(local)

uses_recovered_clock

!uses_recovered_clock

WAIT_ADJACENT

(!mr_training_enable + isl_ready) *
~~adjacent_isl_ready *~~
~~adjacent_remote_rts~~

SWITCH_CLOCK

USE_TX_CLOCK(recovered)
start forward_rts_timer

forward_rts_timer_done

TX_CLOCK_READY

isl_ready +
!mr_training_enable

FORWARD_RTS

local_rts ⇐ true
~~training_status ⇐ READY~~

Figure 178B–9—RTS update state diagram

Operation without training (mr_training_enable=false)

Slides to cover changes for local pattern mode support are provided in the backup.

Review of the proposed solution and feedback is appreciated.

Modifications to D2.1

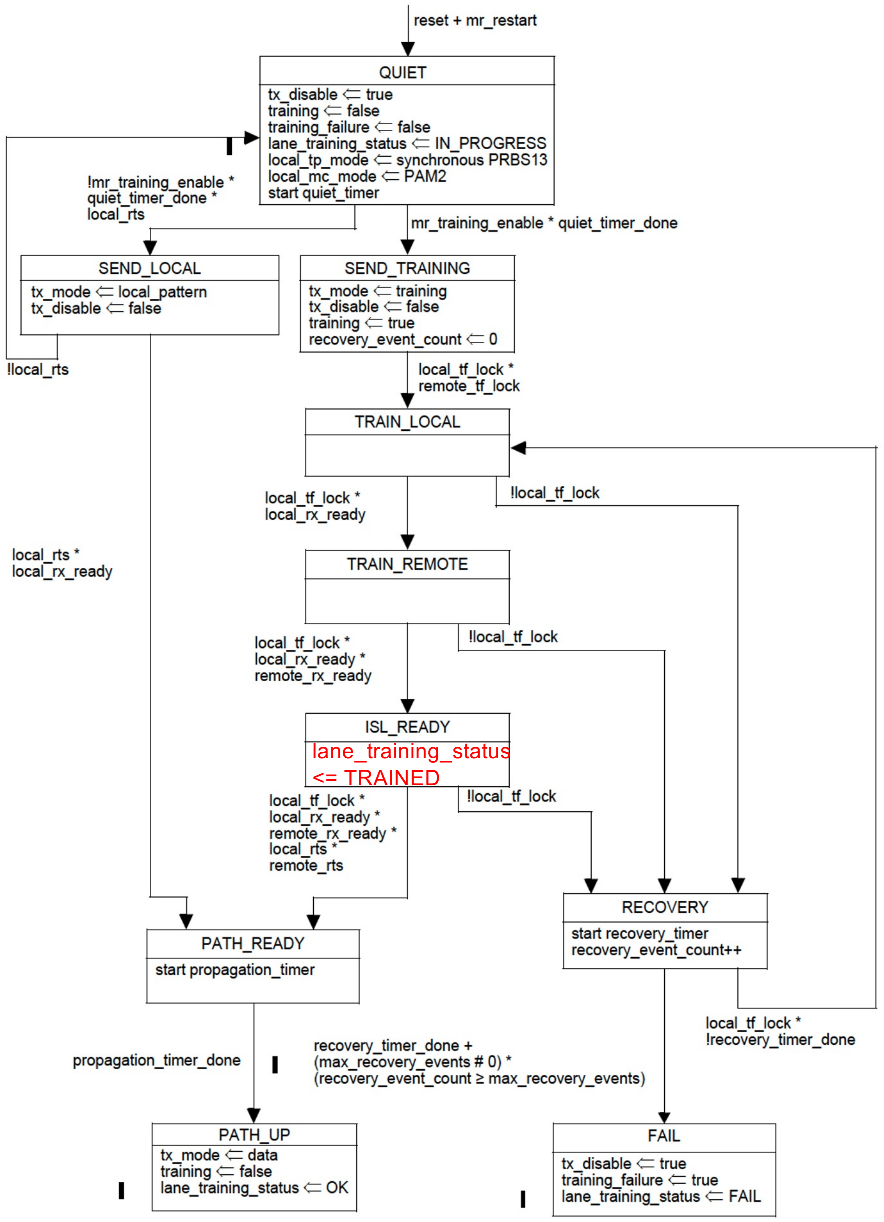
Following slides summarize the changes.

Update (1 / 3)

Changes to Figure 178B-10 highlighted in red.

Define the states of training_status in 178B.7.2.1 to be as follows:

FAIL: Any lane_training_status is FAIL
OK: All lane_training_status are OK
READY: remote_rts is true AND all lanes have lane_training_status of TRAINED or OK with at least one lane being TRAINED.
IN_PROGRESS: At least one lane has lane_training_status of IN_PROGRESS and no lanes have lane_training_status of FAIL.



reset + mr_restart

QUIET
tx_disable ⇐ true
training ⇐ false
training_failure ⇐ false
lane_training_status ⇐ IN_PROGRESS
local_tp_mode ⇐ synchronous PRBS13
local_mc_mode ⇐ PAM2
start quiet_timer

!mr_training_enable *
quiet_timer_done *
local_rts

mr_training_enable * quiet_timer_done

SEND_LOCAL
tx_mode ⇐ local_pattern
tx_disable ⇐ false

!local_rts

SEND_TRAINING
tx_mode ⇐ training
tx_disable ⇐ false
training ⇐ true
recovery_event_count ⇐ 0

local_tf_lock *
remote_tf_lock

TRAIN_LOCAL

local_tf_lock *
local_rx_ready

!local_tf_lock

local_rts *
local_rx_ready

TRAIN_REMOTE

local_tf_lock *
local_rx_ready *
remote_rx_ready

!local_tf_lock

ISL_READY
lane_training_status
<= TRAINED

local_tf_lock *
local_rx_ready *
remote_rx_ready *
local_rts *
remote_rts

!local_tf_lock

RECOVERY
start recovery_timer
recovery_event_count++

PATH_READY
start propagation_timer

local_tf_lock *
!recovery_timer_done

propagation_timer_done

recovery_timer_done +
(max_recovery_events # 0) *
(recovery_event_count ≥ max_recovery_events)

PATH_UP
tx_mode ⇐ data
training ⇐ false
lane_training_status ⇐ OK

FAIL
tx_disable ⇐ true
training_failure ⇐ true
lane_training_status ⇐ FAIL

Figure 178B–10—Training control state diagram

Update (2 / 3)

Changes to Figure 178B-9 indicated in red.

Remove adjacent_remote_rts and adjacent_isl_ready variables definitions and add adjacent_intf_rx_ready as follows:

adjacent_intf_rx_ready

Boolean variable that is derived from the received SIGNAL_OK primitive on the service interface. It is set to TRUE when SIGNAL_OK is either READY or OK and FALSE otherwise.

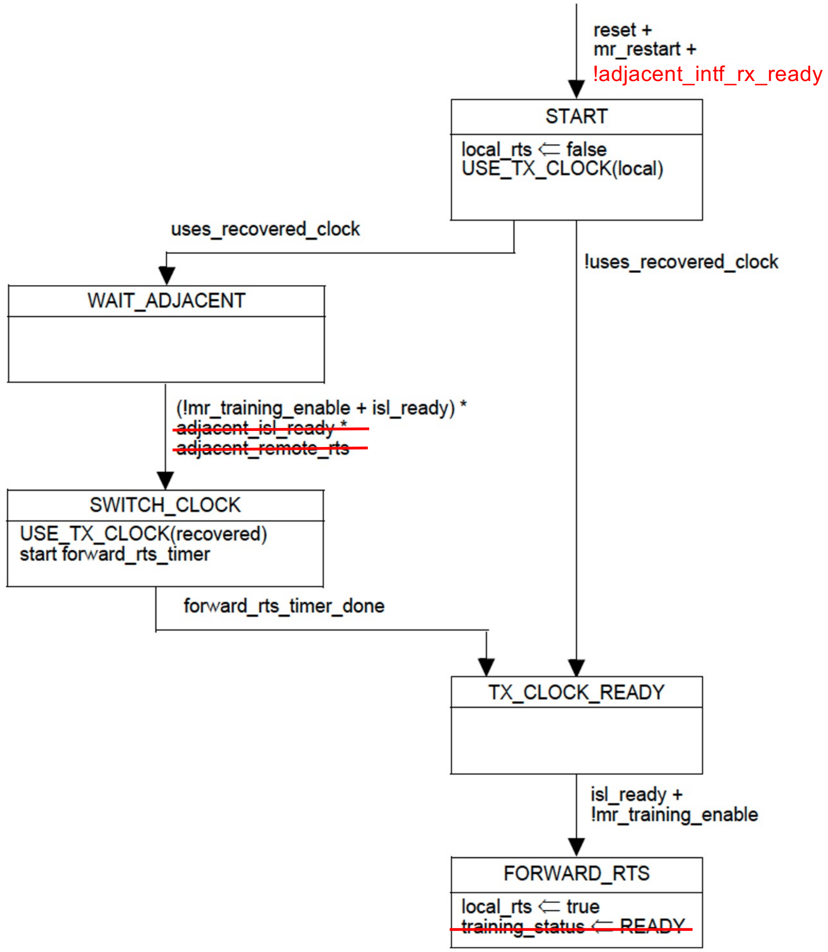


Figure 178B–9—RTS update state diagram

Update (3 / 3)

Add the following definition into 178B.3:

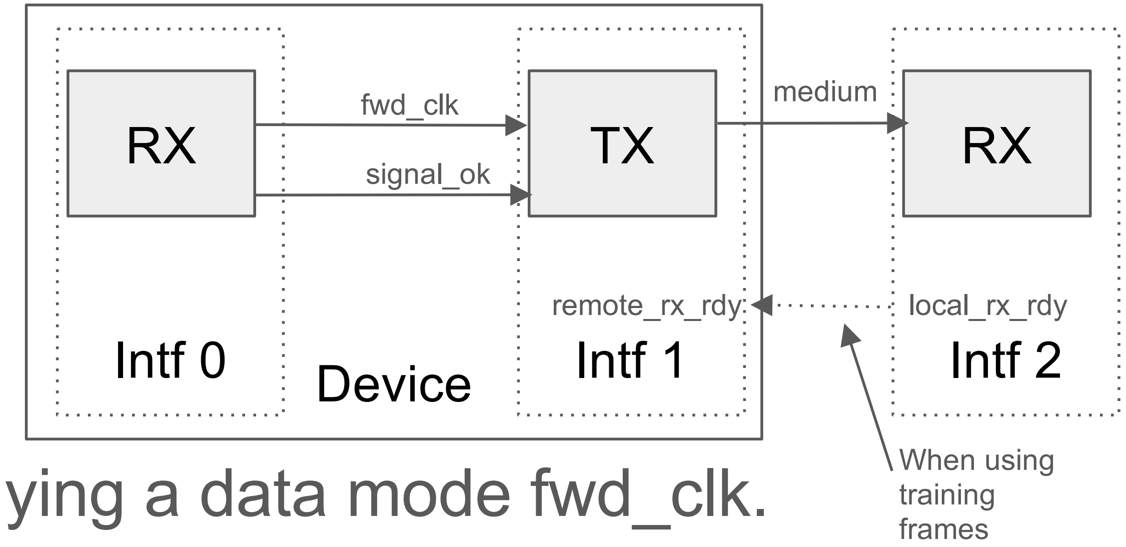
Adjacent interface:

Interface whose service interface is connected (directly or indirectly) with this interface. There may be multiple sub-layers between the two ILT interfaces. For example, in Figure 178B-1 the first AUI component above (left) of the PHY_XS and the first AUI component below (right) of the PCS (for the Physical Layer on the left) are adjacent interfaces of each other.

Thank you

Back up

Flow of the start-up process



When using training frames, an interface has information about the peer interface state, the adjacent interface and its own state.

This permits starting training with the peer interface before the adjacent interface is supplying a data mode fwd_clk.

When using local pattern, the flow of information is unidirectional across the medium; so the interface must wait to send data until the TX is running on its data mode clock as we only have two states “squelch” and “signal” (FAIL / OK).

Path start-up cascades the RTS indicator from RS end down the path. Training frames use the Continue Training field; local pattern we unsquelch.

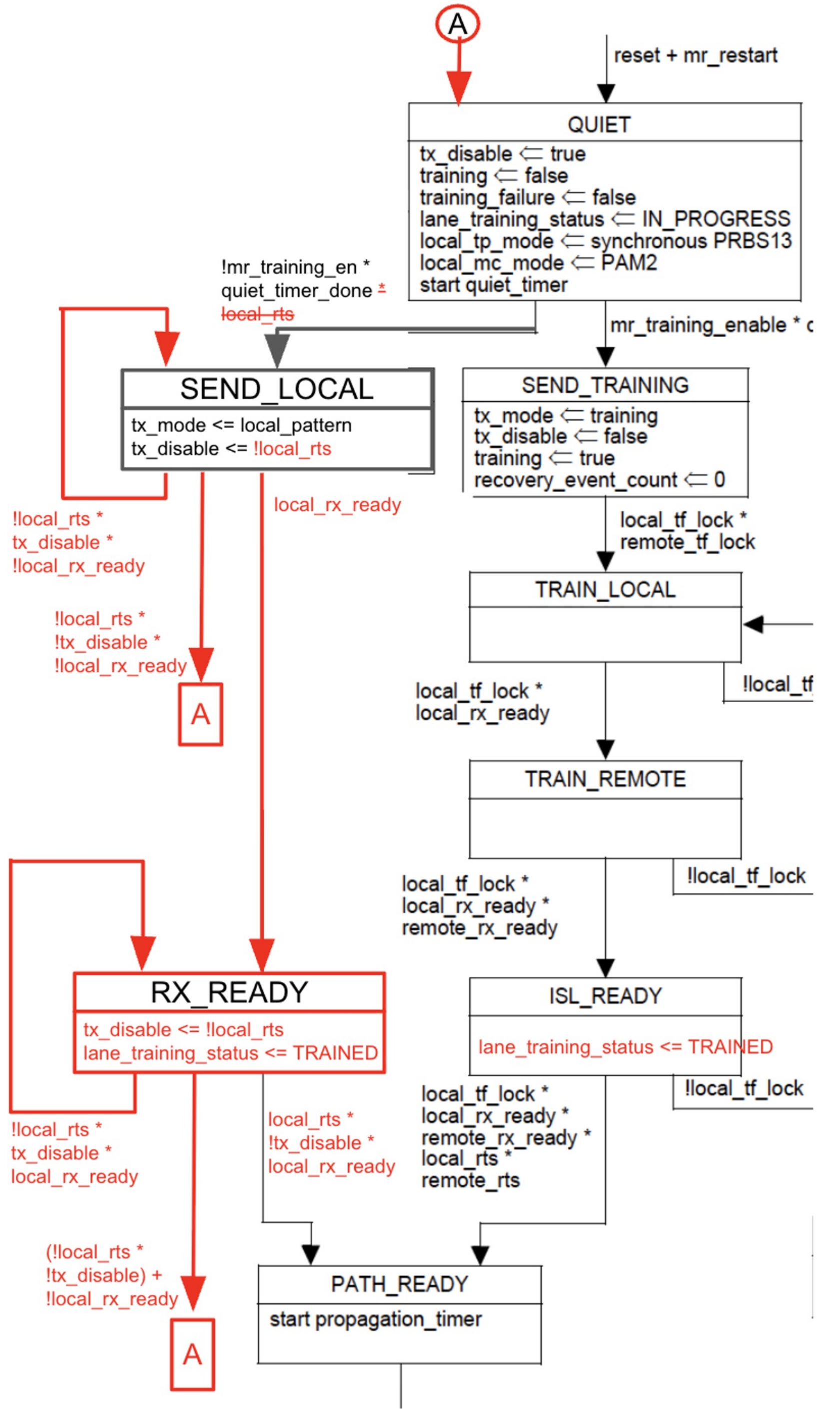
What about local pattern mode?

For mr_training_en = false:

In local pattern mode Tx and Rx are disjoint - one can be ready while other is still “off”. The Tx needs to wait until it’s on mission clock (local_rts = true [item 1,2,3]) to begin transmitting, prior to peer Rx running adaptation. The Rx needs to adapt as soon as it sees data, and set lane_training_status to TRAINED when ready, which indicates the adjacent interface Tx may now swap to mission clock.

The following are the updates to Figure 178B-10 shown to right:

- Ensure Tx is squelched for at least the quiet timer duration. Then support either Tx or Rx completing first.
- Unsquelch Tx once its swapped to mission clock (local_rts = true)
- Set lane_training_status to TRAINED when Rx is ready
- Monitor if local_rts changes from true to false, go back to QUIET to squelch for a minimum period of time.
- Monitor if local_rx_ready changes from true to false, go back to QUIET to squelch for a minimum period of time.



Modifications to D2.1

Following slides summarize the changes for both modes of training (training frames and local pattern).

Update (1 / 3)

Changes to Figure 178B-10 highlighted in red.

Define the states of training_status in 178B.7.2.1 to be as follows:

FAIL: Any lane_training_status is FAIL
OK: All lane_training_status are OK
READY: remote_rts is true AND all lanes have lane_training_status of TRAINED or OK with at least one lane being TRAINED.
IN_PROGRESS: At least one lane has lane_training_status of IN_PROGRESS and no lanes have lane_training_status of FAIL.

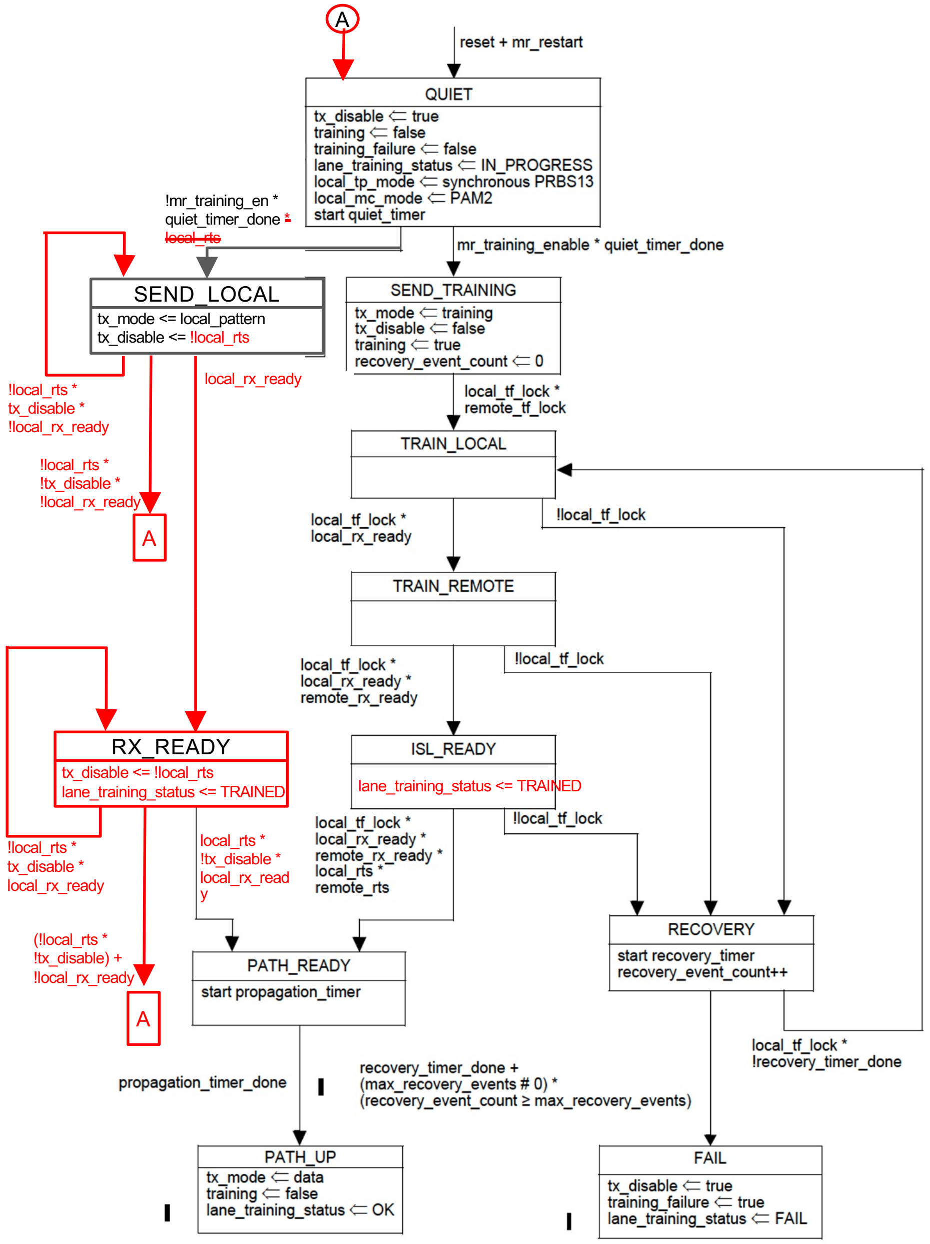


Figure 178B–10—Training control state diagram

Update (2 / 3)

Changes to Figure 178B-9 indicated in red.

Remove adjacent_remote_rts and adjacent_isl_ready variables definitions and add adjacent_intf_rx_ready as follows:

adjacent_intf_rx_ready

Boolean variable that is derived from the received SIGNAL_OK primitive on the service interface. It is set to TRUE when SIGNAL_OK is either READY or OK and FALSE otherwise.

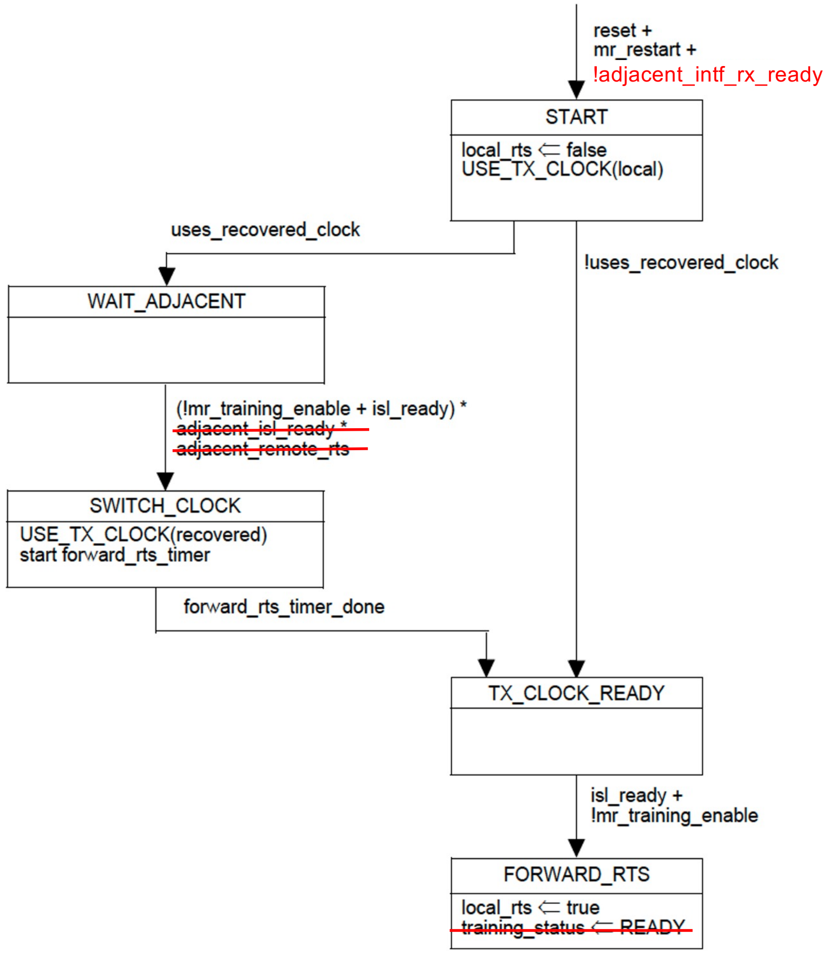


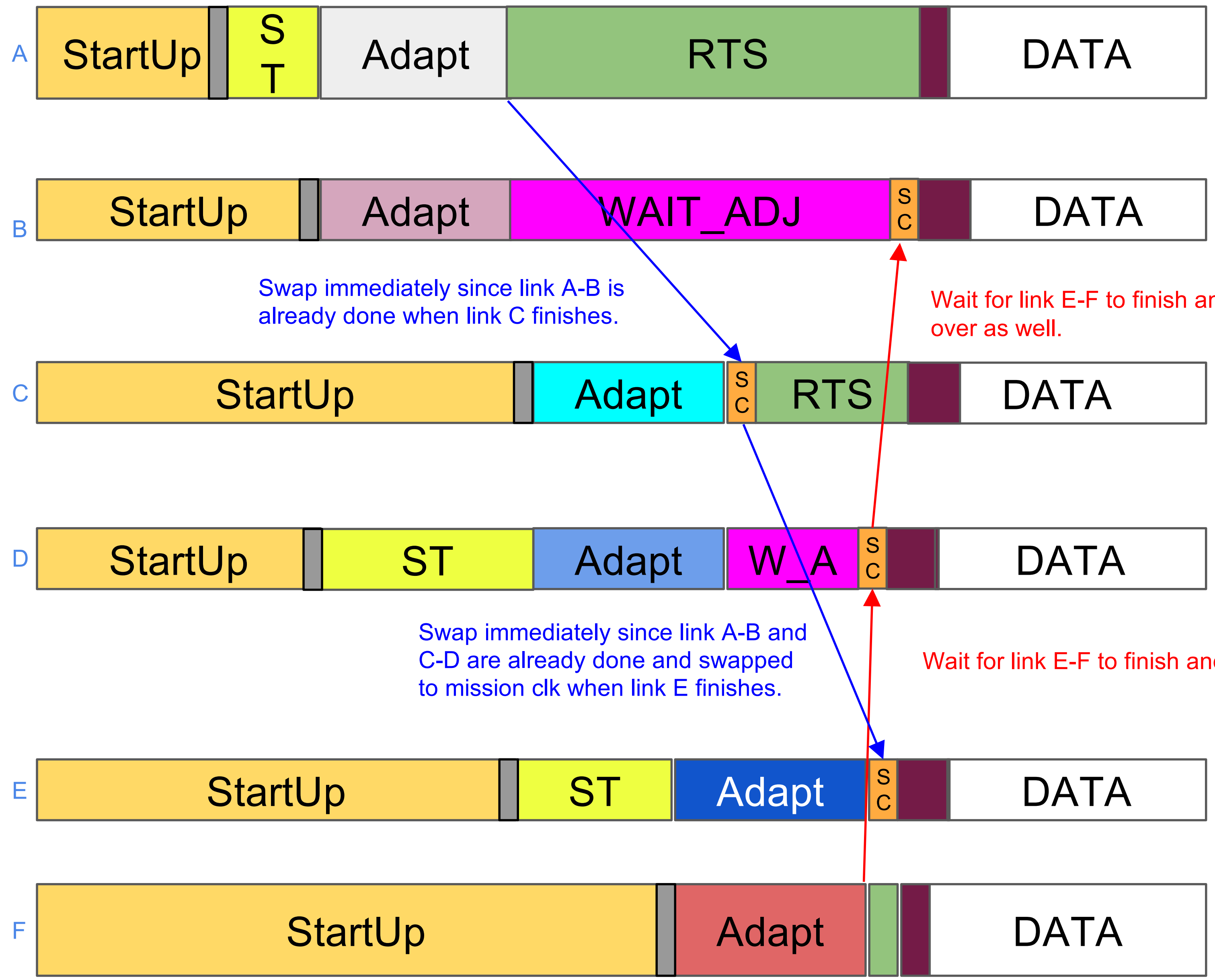
Figure 178B–9—RTS update state diagram

Update (3 / 3)

Add the following definition into 178B.3:

Adjacent interface:

Interface whose service interface is connected (directly or indirectly) with this interface. There may be multiple sub-layers between the two ILT interfaces. For example, in Figure 178B-1 the first AUI component above (left) of the PHY_XS and the first AUI component below (right) of the PCS (for the Physical Layer on the left) are adjacent interfaces of each other.

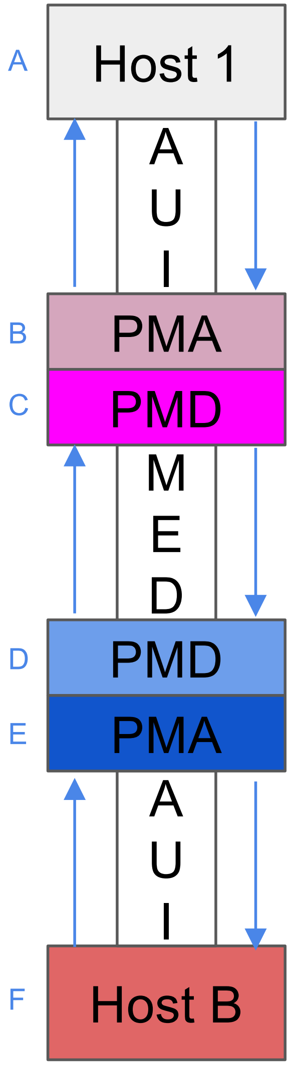


Swap immediately since link A-B is already done when link C finishes.

Wait for link E-F to finish and for D to swap over as well.

Swap immediately since link A-B and C-D are already done and swapped to mission clk when link E finishes.

Wait for link E-F to finish and then swap.


A
Host 1
A
U
I
B
PMA
C
PMD
M
E
D
D
PMD
E
PMA
A
U
I
F
Host B