AUI Architecture and ILT signaling

Addressing Draft 1.1 comment #516, #508, and perhaps others

Matt Brown, Alphawave Semi
Adee Ran, Cisco
Leon Bruckman, Nvidia

Supporters

Howard Heck, Intel

Introduction

- The AUI functionality has evolved drastically in 802.3dj compared with previous generations approaching that of a PMD.
- Introduction of ILT requires more sophisticate signaling between a PMD, AUI component, with many other sublayers.
- The presentation proposes a formal architecture representations of the AUI and subcomponents and proposes update to address the ILT signaling.

Comments

Cl 176E SC 176E.3 P695 L16 # 516

Brown, Matt Alphawave Semi

Comment Type T *Comment Status* X

The AUI-C2M compoent is defined as being "functionally equivalent to a corresponding n-lane PMD specified in Clause 179" and includes the same ILT. However, for the AUI-C2M the functional architecture, like the PMD, including the channel, the component at each end, and the abstract service interface signaling are never defined.

SuggestedRemedy

Define a complete architecture schema for the AUI-C2M as follows:
PMA service interface (above the AUI)
AUI Component
AUI Channel
AUI Component
PMA service interface (below the AUI)
Implement similarly for AUI-C2C in Annex 176D.
A presentation with a more complete proposal will be provided.

Proposed Response *Response Status* O

176E.3 Functional specification

A 200 Gb/s per lane AUI-C2M is a bidirectional data path. Each direction of a 200GAUI-1 C2M, 400GAUI-2 C2M, 800GAUI-4 C2M or 1.6TAUI-8 C2M data path contains one, two, four, or eight lanes, respectively.

A 200 Gb/s per lane AUI-C2M is specified in terms of a host and a module. The host and the module both contain components that implement the interface, referred to as C2M components.

An *n*-lane C2M component is functionally equivalent to a corresponding *n*-lane PMD specified in Clause 179 (see 179.8), and implements the same inter-sublayer service interface (see 179.4), using PAM4 signaling at a nominal signaling rate of 106.25 GBd on each lane.

Cl 176A SC 176A.11.2.1 P642 L46 # 508

Brown, Matt Alphawave Semi

Comment Type T *Comment Status* X

The editor's note points out that the location of the Figure 176A-6 state diagram needs to be specified. Given that there is one per interface and since the ILT function is part of the PMD or AUI component the location is implicit.

SuggestedRemedy

Delete the editor's note.

Proposed Response *Response Status* O

Figure 176A–6—RTS update state diagram

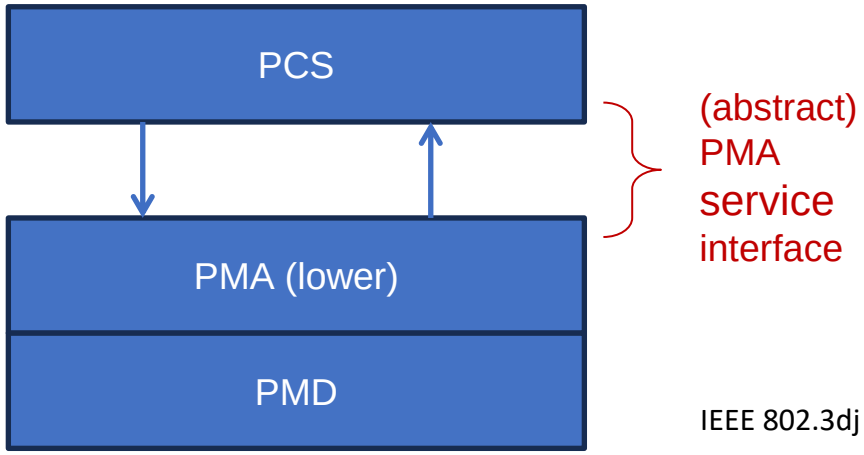
Editor's note: *Need to define where is the RTS state diagram implemented, or leave it open for the implementor's decision*

Part I:
AUI architecture

What is an AUI? (1)

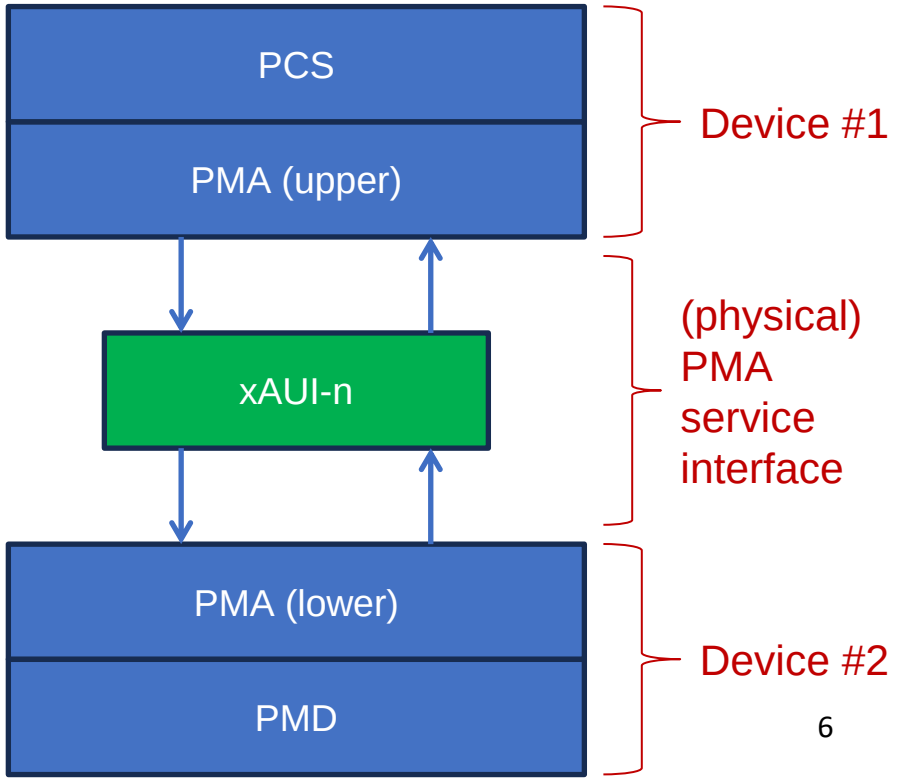
1.4.110 200 Gb/s Attachment Unit Interface (200GAUI-n): A physical instantiation of the PMA service interface to extend the connection between 200 Gb/s capable PMAs over n lanes, used for chip-to-chip or chip-to-module interconnections. Two widths of 200GAUI-n are defined: an eight-lane version (200GAUI-8), and a four-lane version (200GAUI-4). (See IEEE Std 802.3, Annex 120B and Annex 120C for 200GAUI-8, or Annex 120D and Annex 120E for 200GAUI-4.)

1.4.145 400 Gb/s Attachment Unit Interface (400GAUI-n): A physical instantiation of the PMA service interface to extend the connection between 400 Gb/s capable PMAs over n lanes, used for chip-to-chip or chip-to-module interconnections. Two widths of 400GAUI-n are defined: a sixteen-lane version (400GAUI-16), and an eight-lane version (400GAUI-8). (See IEEE Std 802.3, Annex 120B and Annex 120C for 400GAUI-16, or Annex 120D and Annex 120E for 400GAUI-8.)



xAUI-n is a physical instantiation of the PMA service interface.

In other words, it permits sublayers with a Physical Layer implementation to be implemented on separate devices connected by physical interconnect between.



What is an AUI? (2)

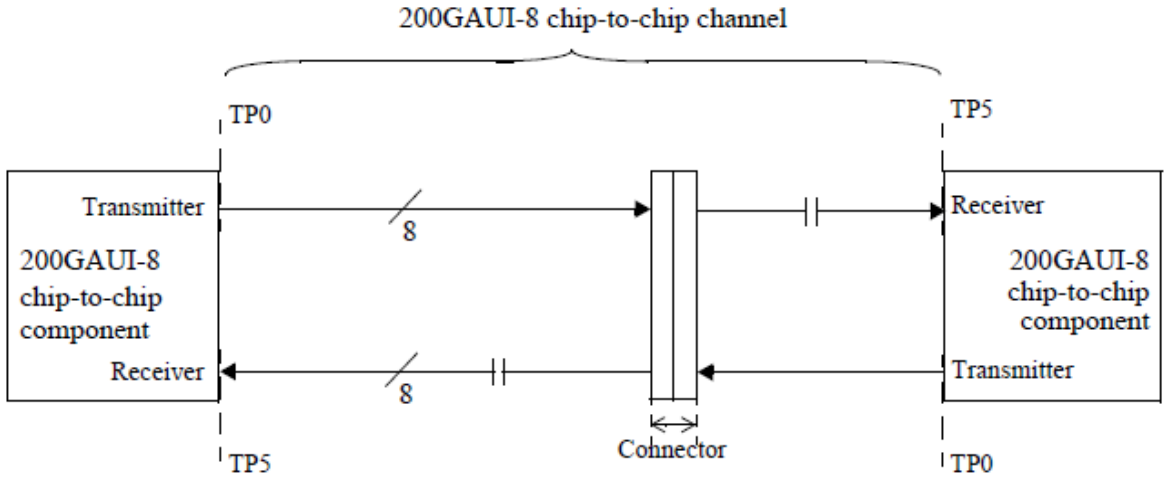


Figure 120B–3—Typical 200GAUI-8 chip-to-chip application

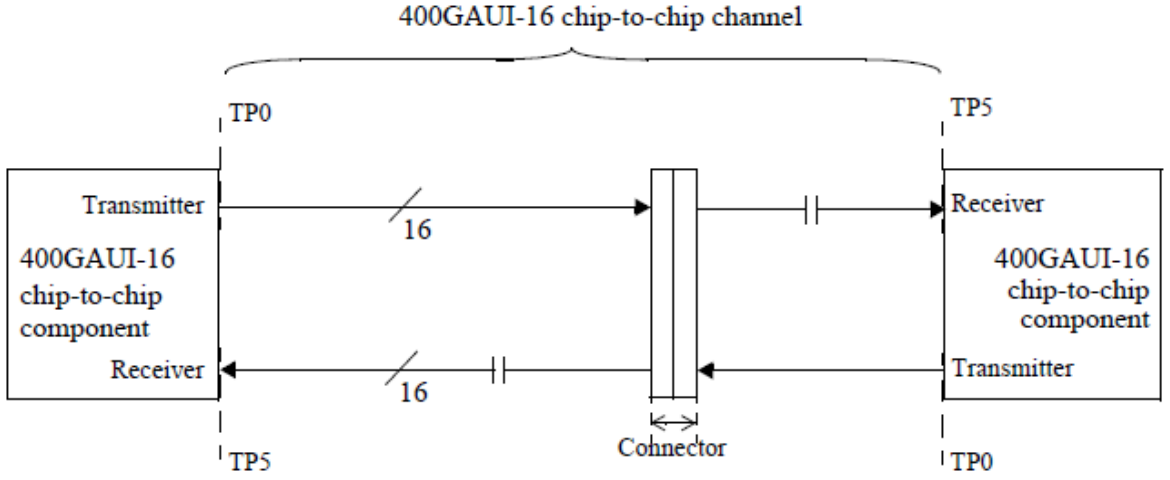
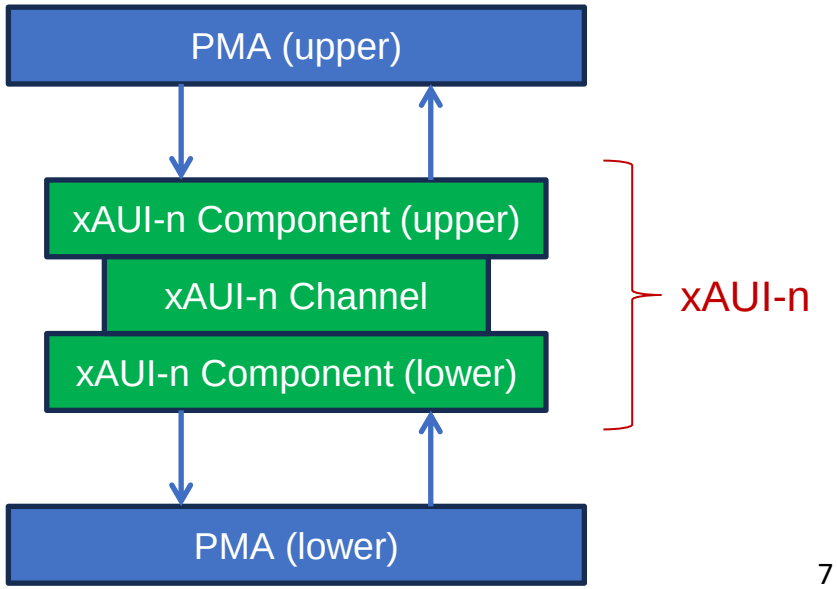


Figure 120B–4—Typical 400GAUI-16 chip-to-chip application

In the defining annexes, an xAUI-n is composed of an xAUI-n component on each end and an xAUI-n channel in between.

The components are defined by electrical characteristics at the outputs and inputs that attach to the channel.

The channel is defined by the electrical characteristics at and between the terminals at each end.



Where was the AUI? (1)

120.5.6 Signal drivers

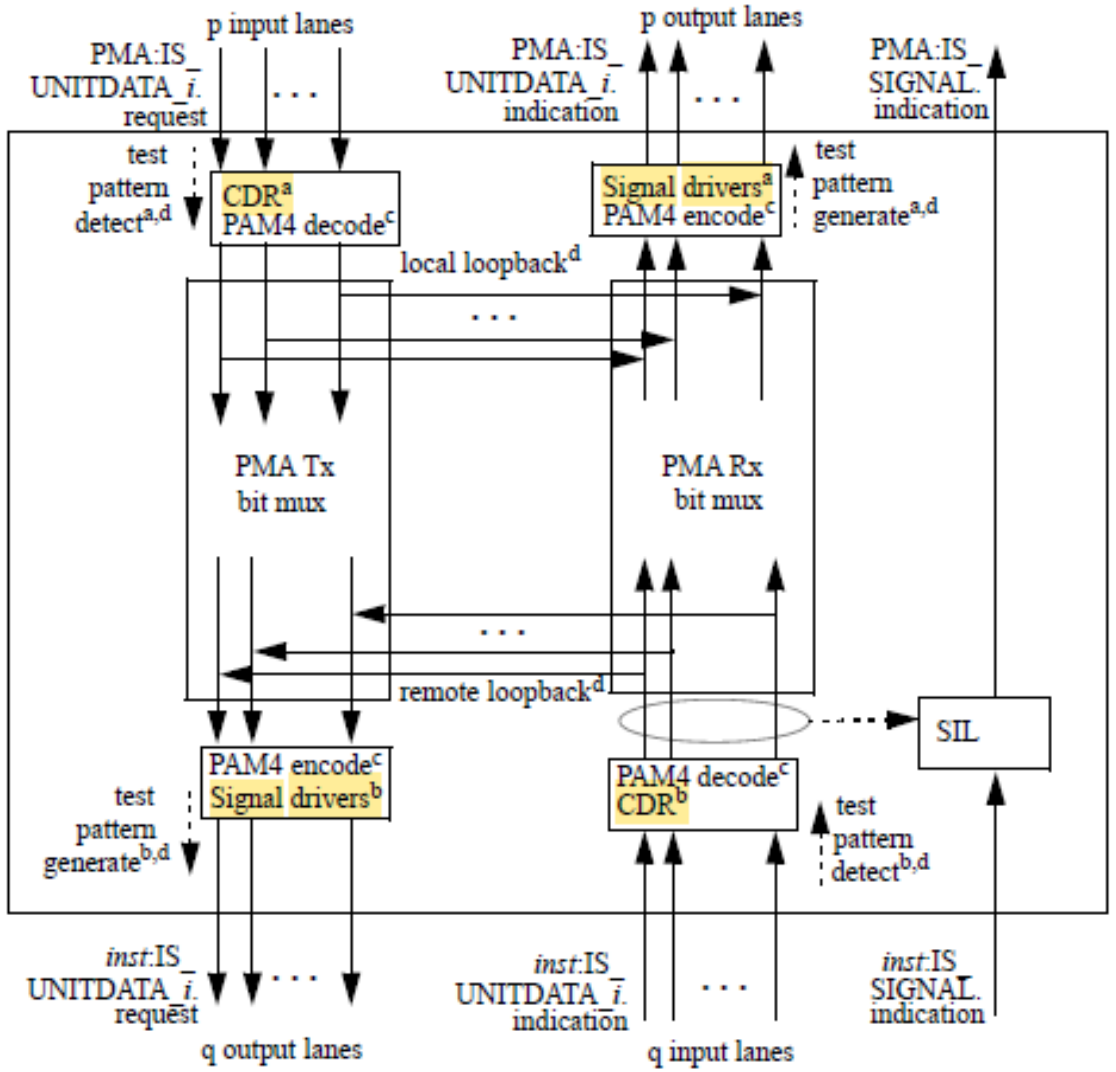
For cases where the interface between the PMA client and the PMA, or between the PMA and the sublayer below the PMA represent a physically instantiated interface, the PMA provides electrical signal drivers for that interface. The electrical and jitter/timing specifications for these interfaces appear in

- Annex 120B, which specifies the 200GAUI-8 and 400GAUI-16 interfaces for chip-to-chip applications.
- Annex 120C, which specifies the 200GAUI-8 and 400GAUI-16 interfaces for chip-to-module applications.
- Annex 120D, which specifies the 200GAUI-4 and 400GAUI-8 interfaces for chip-to-chip applications.
- Annex 120E, which specifies the 200GAUI-4 and 400GAUI-8 interfaces for chip-to-module applications.

For 200GAUI-8 or 400GAUI-16, the modulation format is NRZ. For 200GAUI-4 or 400GAUI-8, the modulation format is PAM4.

120.4 Service interface below PMA

In the Rx direction, if the symbol is received over a physically instantiated interface (200GAUI-n, 400GAUI-n, or physically instantiated PMD service interface), clock and data are recovered on the lane receiving the symbol. If necessary, PAM4 symbols received on the input lanes are converted to pairs of bits. The bits are routed through the PMA to an output lane toward the PMA client through a process that may demultiplex PCSLs from the input, perform any necessary buffering to tolerate Skew Variation across input lanes, and multiplex PCSLs to output lanes. If necessary, pairs of bits are converted to PAM4 symbols on the output lanes. Each symbol is sent on an output lane to the PMA client using the PMA:IS_UNITDATA_*k*.indication (*k* not necessarily equal to *i*) primitive at the PMA service interface.



inst PMD, PMA, or PHY_XS, depending on which sublayer is below this PMA
SIL Signal Indication Logic

[a] If physically instantiated interface (200GAUI-n or 400GAUI-n) immediately above this PMA.
[b] If physically instantiated interface (200GAUI-n or 400GAUI-n) immediately below this PMA, or if this is the closest PMA to the PMD.
[c] If number of input or output lanes is 4 for a 200GBASE-R PMA, or 4 or 8 for a 400GBASE-R PMA.
[d] Optional.

Figure 120–5—PMA Functional Block Diagram

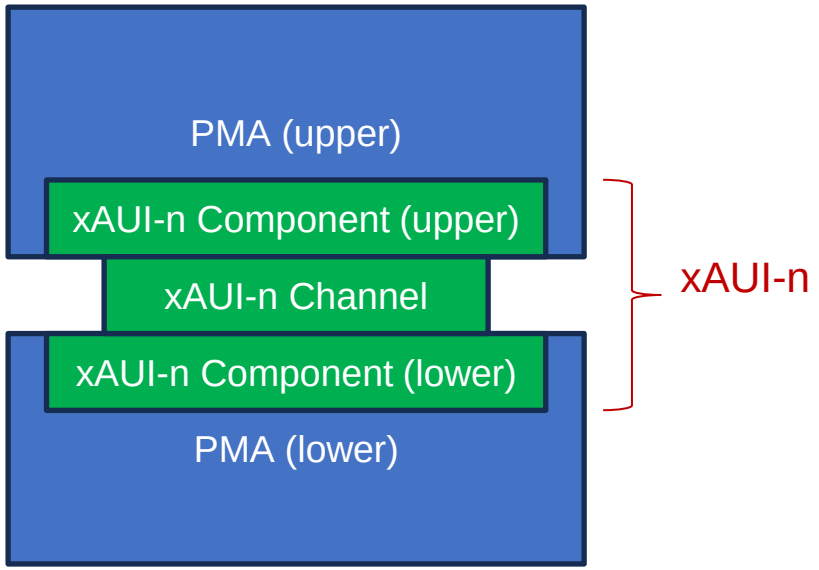
Where was the AUI? (2)

For xAUI-n, defined with 10 Gb/s, 25 Gb/s, 50 Gb/s, and 100 Gb/s the TX signal drivers (for xAUI-n only) and RX CDR (for xAUI-n or PMD with physically instantiated service interface, e.g., PPI) were defined to be part of the PMA.

That implies that the xAUI-n components were part of the PMA.

Note that 802.3dj does not define physical instantiations of the PMD service interface.

This architecture framework worked well for 10 Gb/s and 25 Gb/s where the SERDES was relatively straight forward and (at the time) implemented typically using mixed-signal rather than an integrated DSP engine.



Where is the AUI? (1)

The two AUI defined in 802.3dj, with 200 Gb/s signaling are quite unique from their predecessors:

- They include link training (ILT)
- ILT must be coordinated with other physical instantiations along a network path
- Defined as being equivalent to their PMD counterpart, C2M to CR and C2C to KR.

As a result, the AUI components look and behave a lot like a PMD and likely will be similarly implemented.

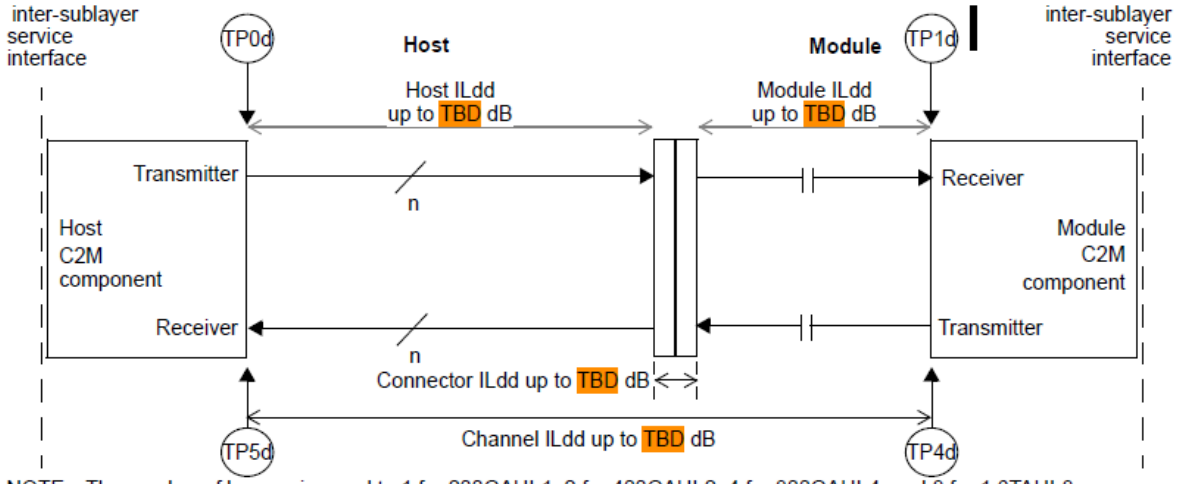
So, it would make sense to model them architecturally the same way.

176E.3 Functional specification

A 200 Gb/s per lane AUI-C2M is a bidirectional data path. Each direction of a 200GAUI-1 C2M, 400GAUI-2 C2M, 800GAUI-4 C2M or 1.6TAUI-8 C2M data path contains one, two, four, or eight lanes, respectively.

A 200 Gb/s per lane AUI-C2M is specified in terms of a host and a module. The host and the module both contain components that implement the interface, referred to as C2M components.

An *n*-lane C2M component is functionally equivalent to a corresponding *n*-lane PMD specified in Clause 179 (see 179.8), and implements the same inter-sublayer service interface (see 179.4), using PAM4 signaling at a nominal signaling rate of 106.25 GBd on each lane.



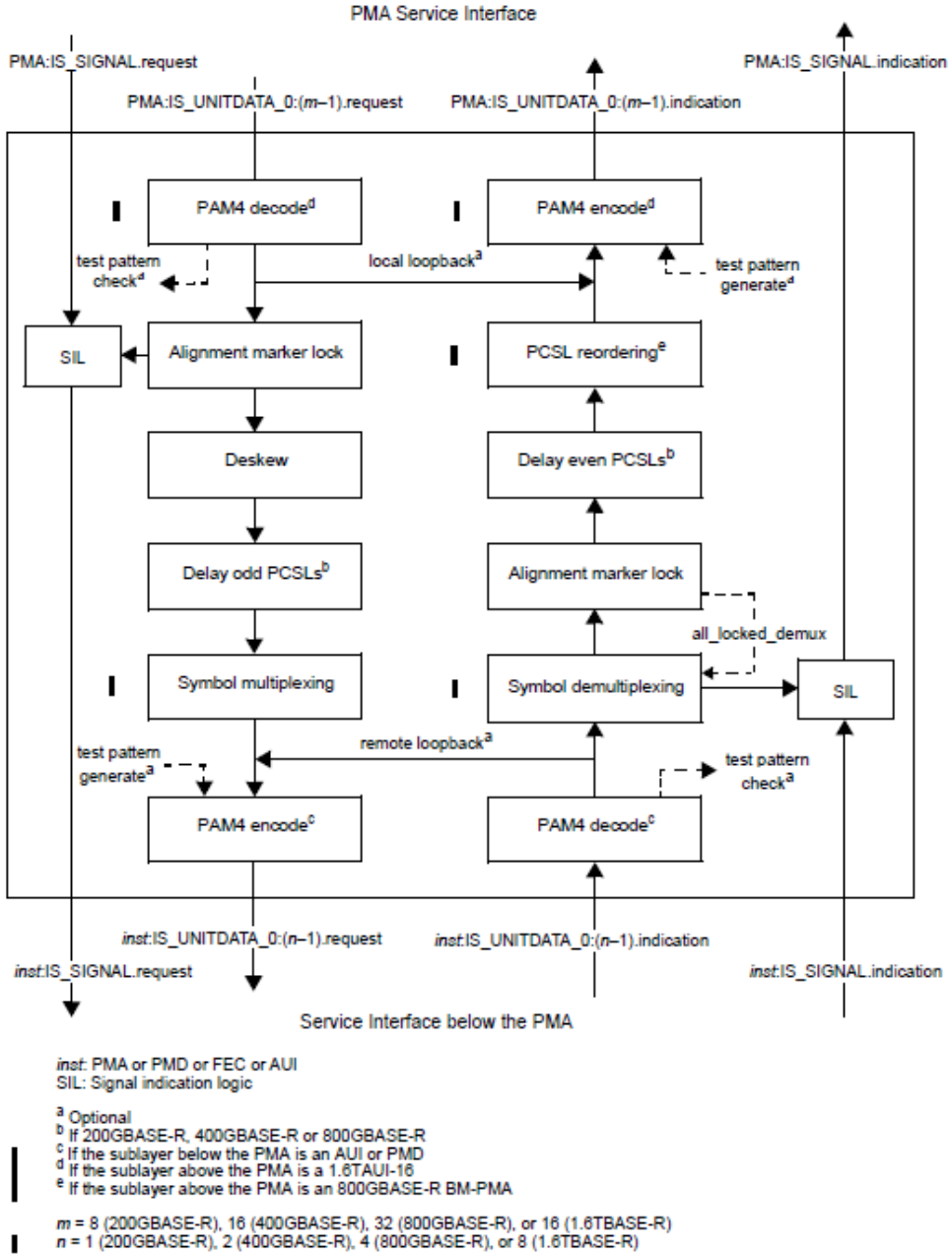
NOTE—The number of lanes n is equal to 1 for 200GAUI-1, 2 for 400GAUI-2, 4 for 800GAUI-4, and 8 for 1.6TAUI-8.

Figure 176E–2—Components of a 200 Gb/s per lane AUI-C2M and insertion loss budget at 53.125 GHz

Where is the AUI? (2)

For 802.3dj, the CDR and signal driver is no longer part of the PMA. Instead, given the AUI components are equivalent to the PMD, these functions are now implicitly within the AUI.

However, these details for the AUI have not been provided in Draft 1.1.



inst: PMA or PMD or FEC or AUI
SIL: Signal indication logic

[a] Optional
[b] If 200GBASE-R, 400GBASE-R or 800GBASE-R
[c] If the sublayer below the PMA is an AUI or PMD
[d] If the sublayer above the PMA is a 1.6TAUI-16
[e] If the sublayer above the PMA is an 800GBASE-R BM-PMA

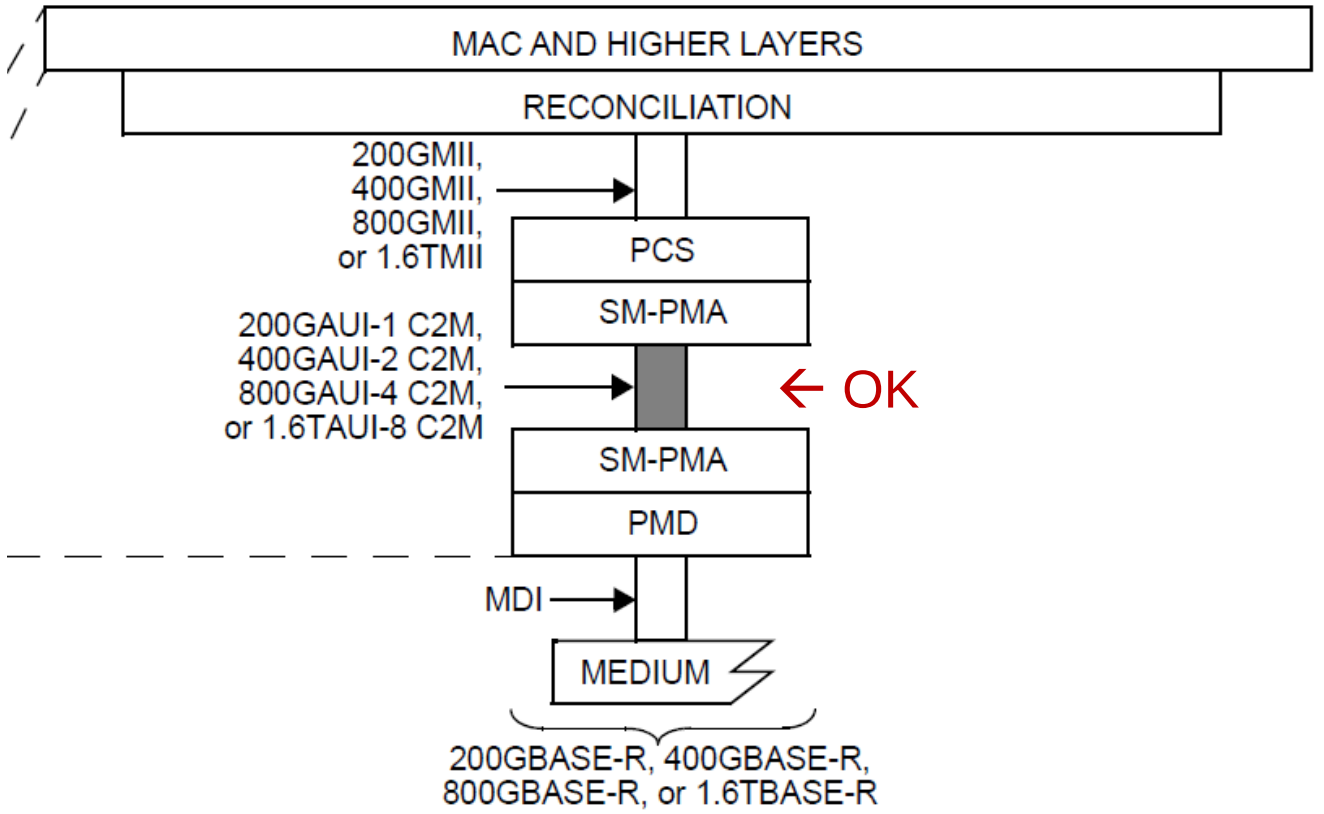
m = 8 (200GBASE-R), 16 (400GBASE-R), 32 (800GBASE-R), or 16 (1.6TBASE-R)
n = 1 (200GBASE-R), 2 (400GBASE-R), 4 (800GBASE-R), or 8 (1.6TBASE-R)

Figure 176–2—200GBASE-R 8:1, 400GBASE-R 16:2, 800GBASE-R 32:4, 1.6TBASE-R 16:8 PMAs functional block diagram

AUI architecture proposed (1)

The xAUI-n can be modelled as an architectural element between a pair of PMAs.

No change for PHY stacks.

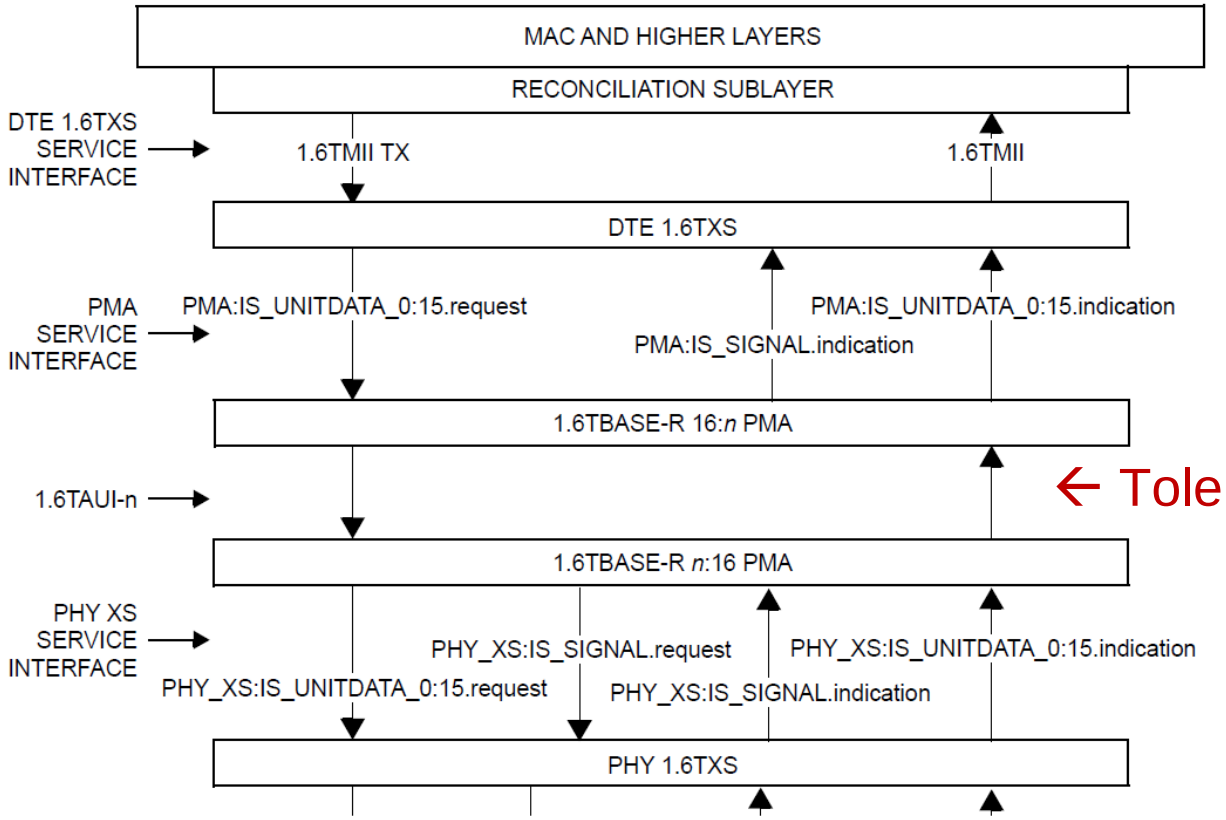


AUI architecture proposed (2)

For general service interface diagrams, it would be nice to depict the xAUI-n like sublayers with service interfaces.

However, then the common diagrams would not be compatible with lower rate (< 200 Gb/s per lane) xAUI-n.

So, leave these diagrams as they are. Maybe we can revisit this later.



← Tolerable.

Figure 174–3—Inter-sublayer service interfaces for a 1.6TBASE-R PHY and 1.6TMII Extender

AUI architecture proposed (3)

However, as noted earlier, an abstract messaging conduit between the xAUI-n and the PMA above and below is required.

A supporting diagram and text should be provided in each xAUI-n annex.

Add the diagram to the right to Annex 176D and 176E in a new subclause defining the service interfaces.

Details of the service interfaces are provided in the second part of this contribution.

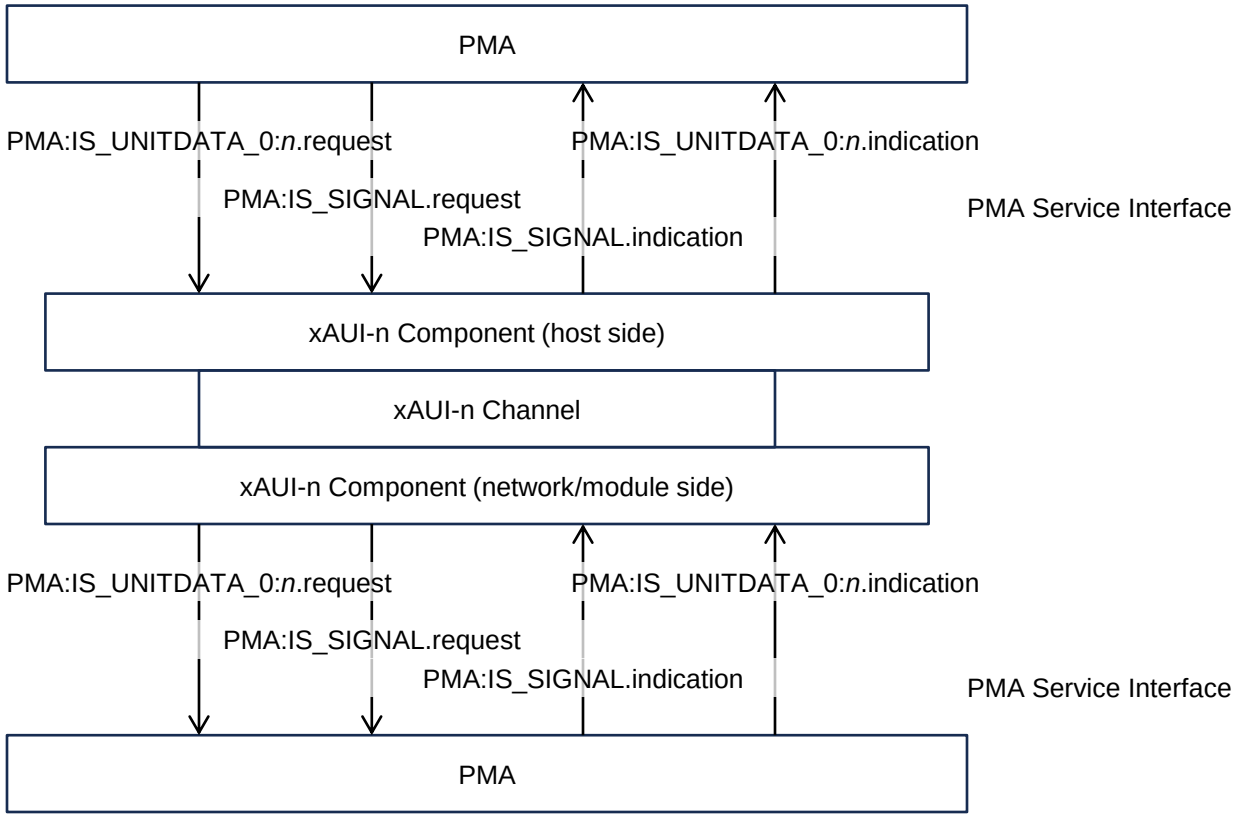


Figure 176D/E-x—Inter-sublayer service interfaces for a 200GAUI-1, 400GAUI-2, 800GAUI-4, 1.6TAUI-4 C2C/C2M

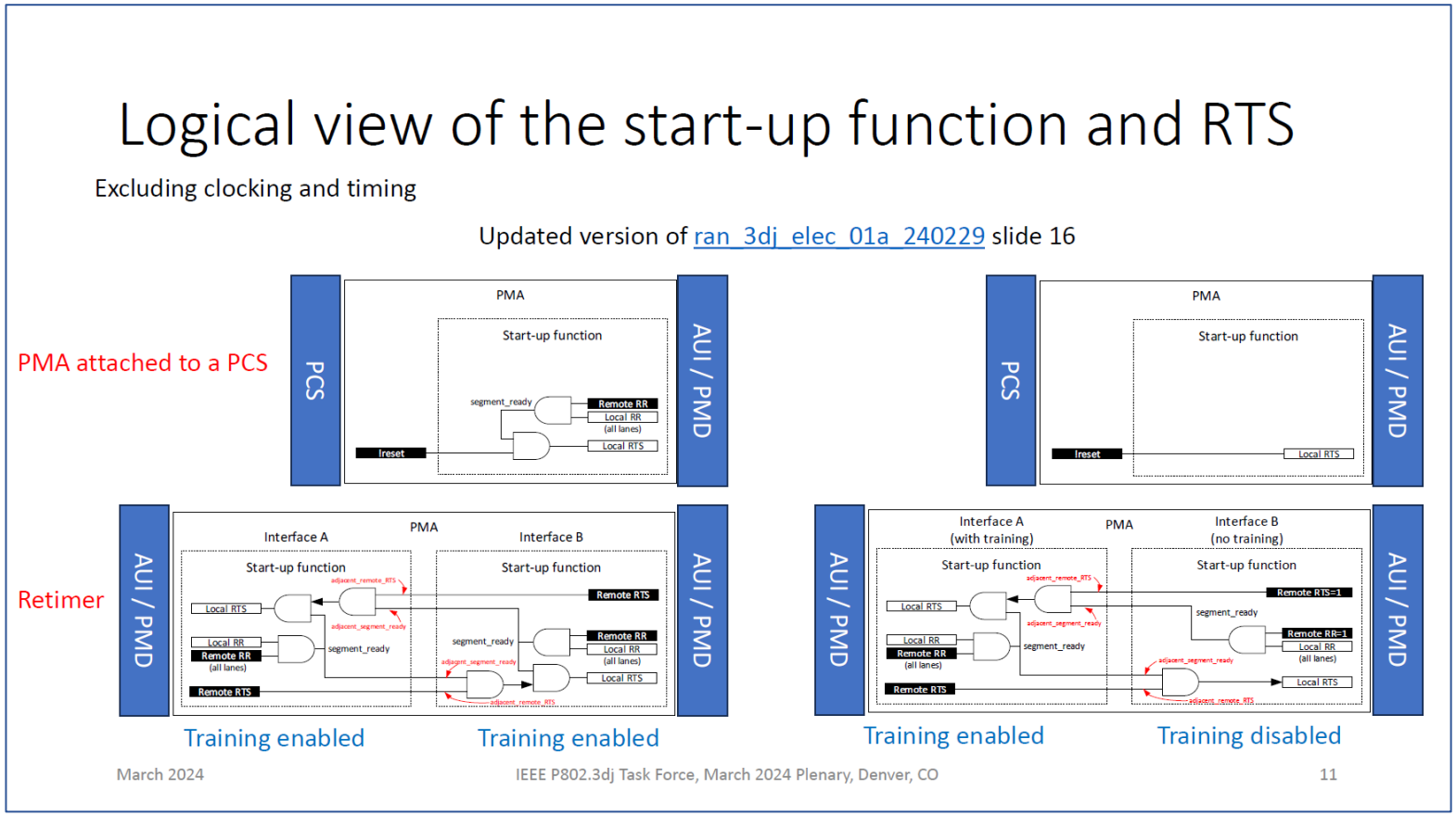
Part II:
ILT signaling

Original ILT concept

The original proposal for ILT it was assumed that the related function would be part of the PMA.

Necessary signaling between interfaces on the same device would occur within a signal sublayer, the PMA.

However, the architecture has evolved such that the ILT function is now part of the PMD or AUI component. A signaling pathway between these elements needs to be defined.



Source: https://www.ieee802.org/3/dj/public/24_03/ran_3dj_04_2403.pdf

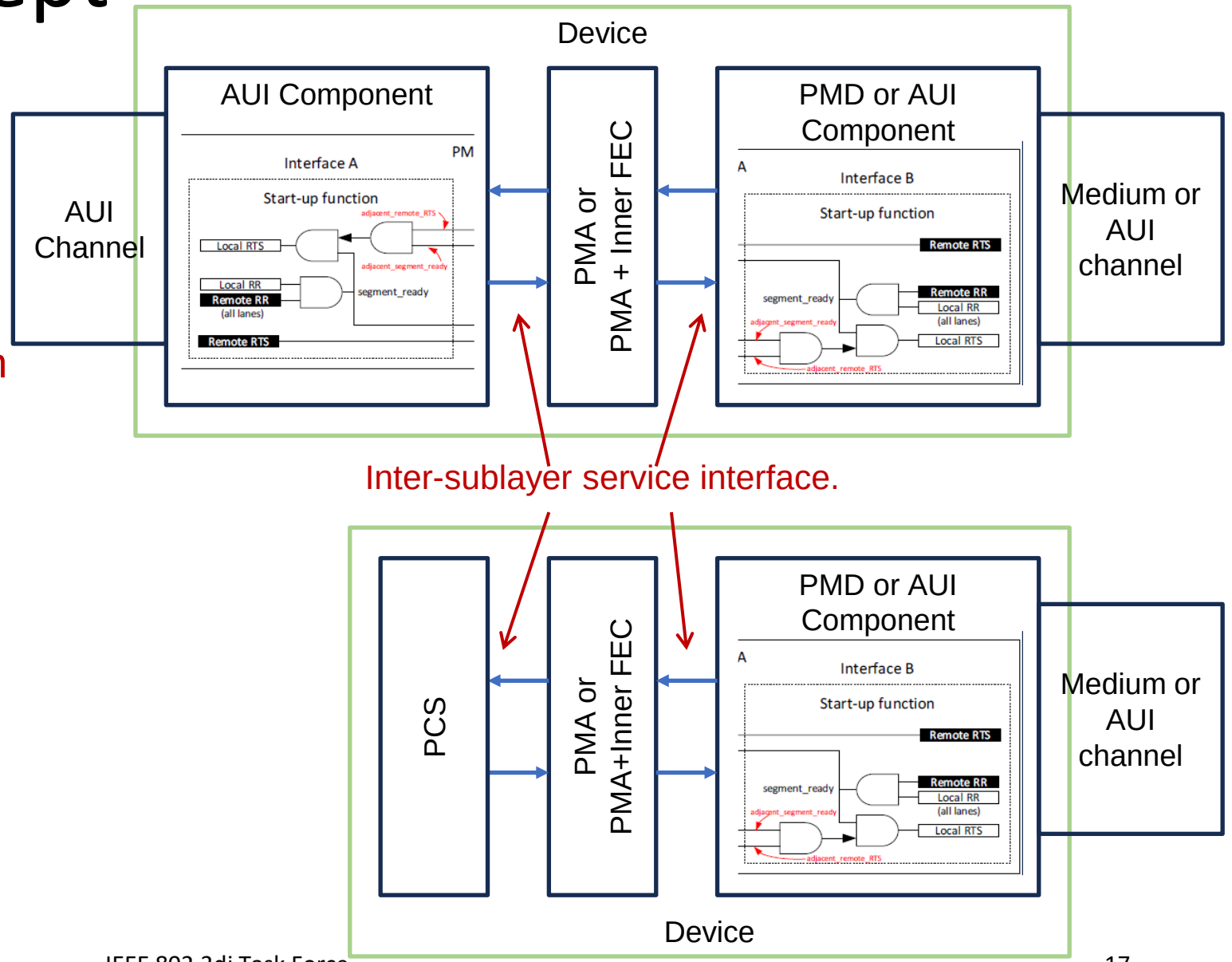
Current ILT concept

The ILT function is now part of the PMD and the AUI component.

Interfaces on the same device are within separated sublayers.

It is not possible to define or depict the signaling between interfaces as signals within a sublayer since the ILT functions are in separate sublayers.

Instead, 802.3 uses service interfaces for this purpose.



Clocking

The ILT function includes provision for careful sequencing of clock source and data source.

It is currently depicted as though the two interfaces are within the same sublayer.

However, as shown on the previous slide there is another sublayer between.

Text and/or diagrams are needed to convey this.

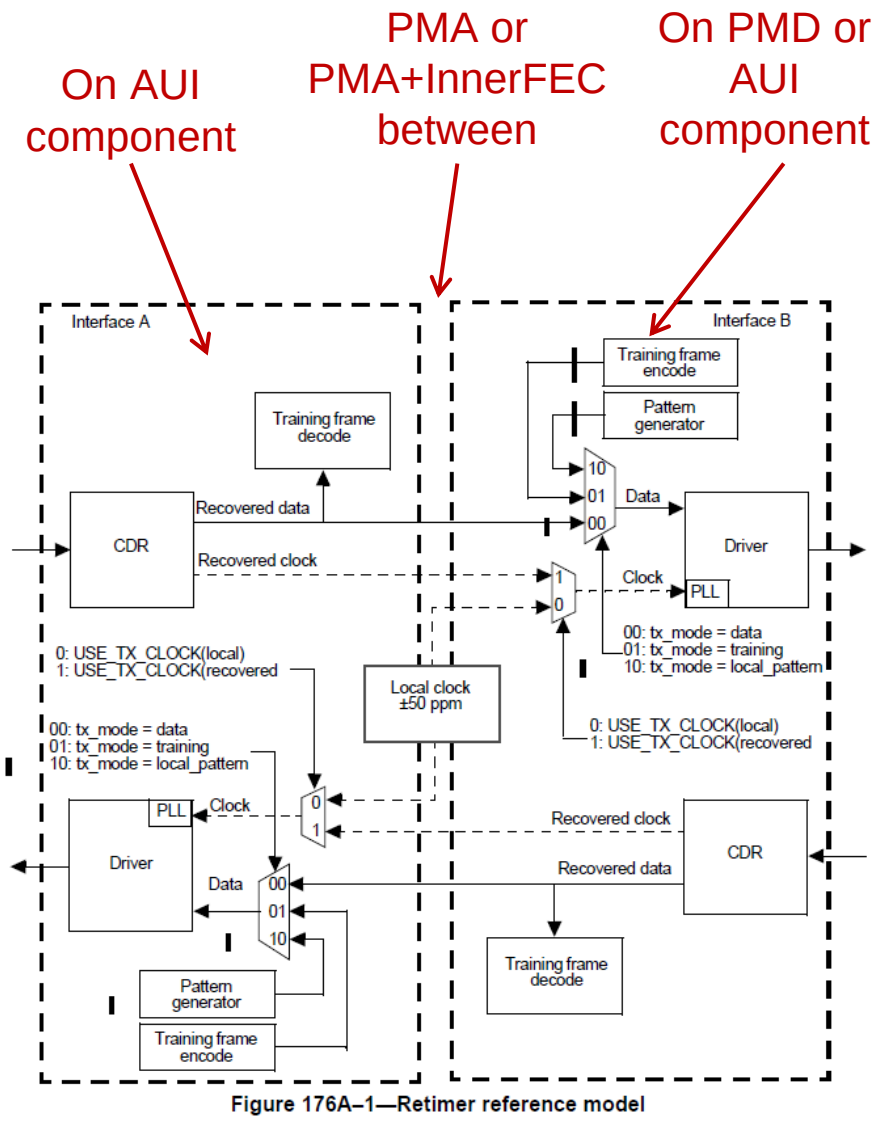


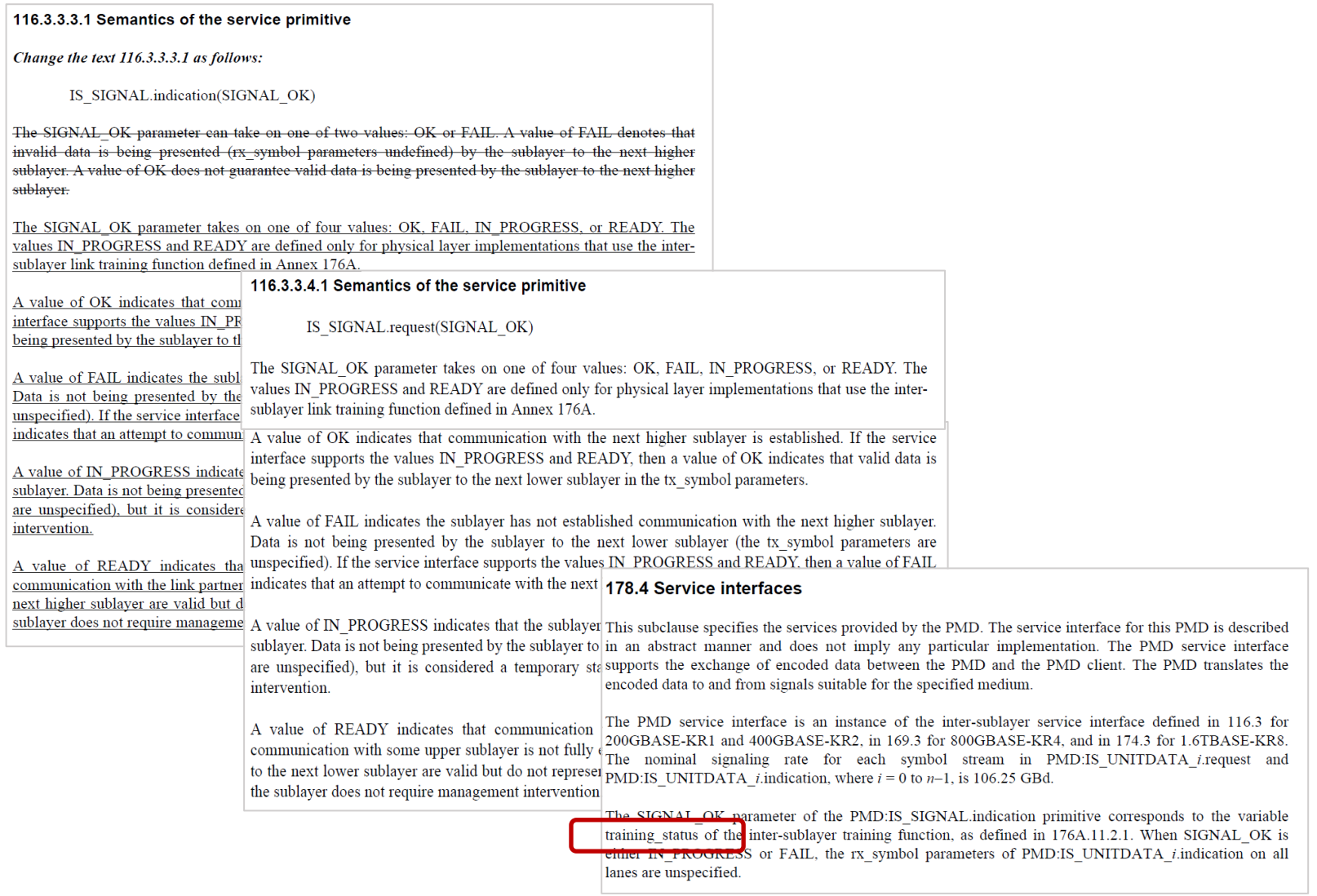
Figure 176A–1—Retimer reference model

Passing ILT states between interfaces (1)

The SIGNAL_OK parameter has been redefined to convey the status of the ILT function on each interface.

SIGNAL_OK is assigned the value of the variable training_status for each of the PMDs.

A similar assignment has not been defined for the AUI components.



116.3.3.3.1 Semantics of the service primitive

Change the text 116.3.3.3.1 as follows:

IS_SIGNAL.indication(SIGNAL_OK)

~~The SIGNAL_OK parameter can take on one of two values: OK or FAIL. A value of FAIL denotes that invalid data is being presented (tx_symbol parameters undefined) by the sublayer to the next higher sublayer. A value of OK does not guarantee valid data is being presented by the sublayer to the next higher sublayer.~~

The SIGNAL_OK parameter takes on one of four values: OK, FAIL, IN_PROGRESS, or READY. The values IN_PROGRESS and READY are defined only for physical layer implementations that use the inter-sublayer link training function defined in Annex 176A.

A value of OK indicates that comm
interface supports the values IN_PR
being presented by the sublayer to tl

A value of FAIL indicates the subl
Data is not being presented by the
unspecified). If the service interface
indicates that an attempt to commun

A value of IN_PROGRESS indicate
sublayer. Data is not being presented
are unspecified), but it is considere
intervention.

A value of READY indicates tha
communication with the link partner
next higher sublayer are valid but d
sublayer does not require manageme

116.3.3.4.1 Semantics of the service primitive

IS_SIGNAL.request(SIGNAL_OK)

The SIGNAL_OK parameter takes on one of four values: OK, FAIL, IN_PROGRESS, or READY. The values IN_PROGRESS and READY are defined only for physical layer implementations that use the inter-sublayer link training function defined in Annex 176A.

A value of OK indicates that communication with the next higher sublayer is established. If the service interface supports the values IN_PROGRESS and READY, then a value of OK indicates that valid data is being presented by the sublayer to the next lower sublayer in the tx_symbol parameters.

A value of FAIL indicates the sublayer has not established communication with the next higher sublayer. Data is not being presented by the sublayer to the next lower sublayer (the tx_symbol parameters are unspecified). If the service interface supports the values IN_PROGRESS and READY, then a value of FAIL indicates that an attempt to communicate with the next

A value of IN_PROGRESS indicates that the sublayer
sublayer. Data is not being presented by the sublayer to
are unspecified), but it is considered a temporary st
intervention.

A value of READY indicates that communication
communication with some upper sublayer is not fully
to the next lower sublayer are valid but do not represe
the sublayer does not require management intervention

178.4 Service interfaces

This subclause specifies the services provided by the PMD. The service interface for this PMD is described in an abstract manner and does not imply any particular implementation. The PMD service interface supports the exchange of encoded data between the PMD and the PMD client. The PMD translates the encoded data to and from signals suitable for the specified medium.

The PMD service interface is an instance of the inter-sublayer service interface defined in 116.3 for 200GBASE-KR1 and 400GBASE-KR2, in 169.3 for 800GBASE-KR4, and in 174.3 for 1.6TBASE-KR8. The nominal signaling rate for each symbol stream in PMD:IS_UNITDATA_*i*.request and PMD:IS_UNITDATA_*i*.indication, where $i = 0$ to $n-1$, is 106.25 GBd.

The SIGNAL_OK parameter of the PMD:IS_SIGNAL.indication primitive corresponds to the variable training_status of the inter-sublayer training function, as defined in 176A.11.2.1. When SIGNAL_OK is either IN_PROGRESS or FAIL, the rx_symbol parameters of PMD:IS_UNITDATA_*i*.indication on all lanes are unspecified.

Passing ILT states between interfaces (2)

The variable training_status is set by the two ILT state machines, defined in Figure 176A-6 and Figure 176A-7.

Based on these state machines the adjacent interface SIGNAL_OK parameter may be interpreted as shown in the table below.

Received SIGNAL_OK value	adjacent_remote_rts	adjacent_isl_ready	Fail?
OK	1	1	No
READY	0	1	No
IN_PROGRESS	0	0	No
FAIL	0	0	Yes

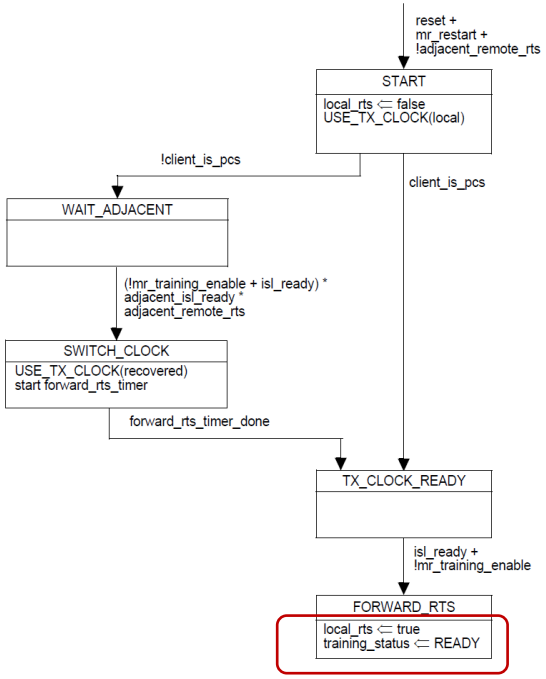


Figure 176A–6—RTS update state diagram

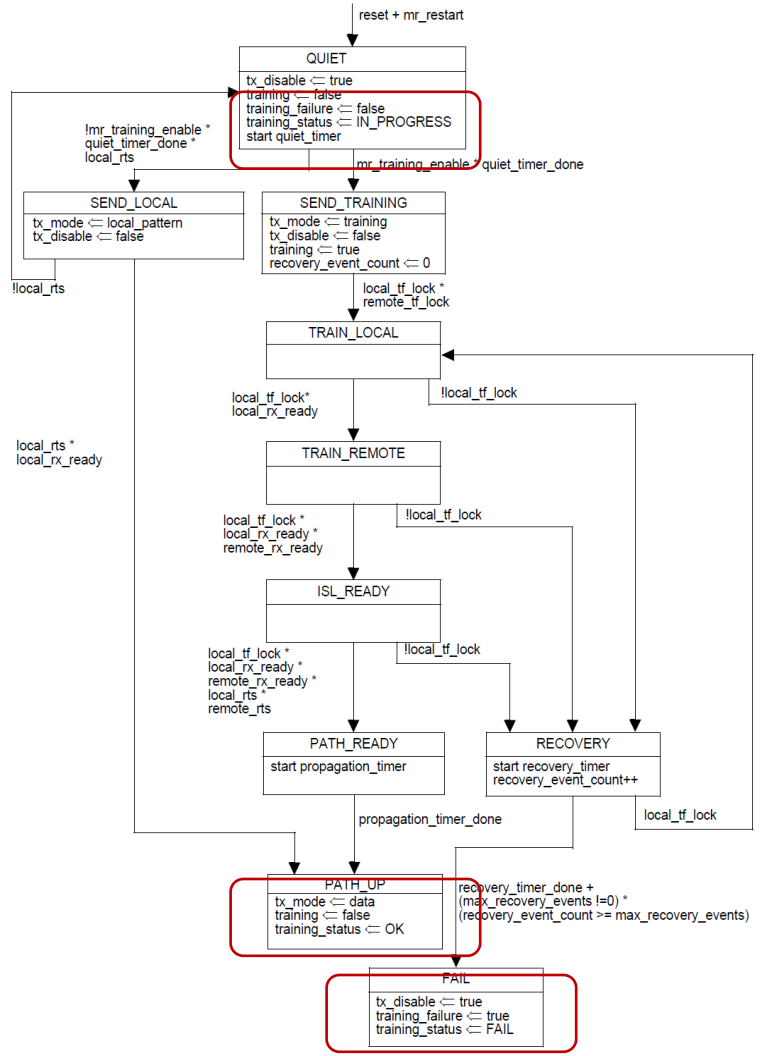


Figure 176A–7—Training control state diagram

Defining AUI service interfaces

The xAUI-n service interfaces should be defined in text.

The service interface, per the definition of an AUI, is the PMA service interface.

However, the interface above and below is subtly different.

Change the text in 176D.1 and 176E.1 and add a new subclause as shown on the right.

In 176D.3…

An n-lane C2C component is functionally equivalent to a corresponding n-lane PMD specified in Clause 178 (see 178.8)~~, and implements the same inter-sublayer service interface (see 179.4),~~ using PAM4 signaling at a nominal signaling rate of 106.25 GBd on each lane. The service interfaces are defined in 176D.x.

In 176E.3…

An n-lane C2M component is functionally equivalent to a corresponding n-lane PMD specified in Clause 179 (see 179.8)~~, and implements the same inter-sublayer service interface (see 179.4),~~ using PAM4 signaling at a nominal signaling rate of 106.25 GBd on each lane. The service interfaces are defined in 176E.x.

Add the following in 176D and 176E

176D/E.x Service interfaces

The PMA above the 200 Gb/s per lane AUI-C2C/C2M is any *m*:1 PMA for 200GAUI-1, *m*:2 PMA for 400GAUI-2, *m*:4 PMA for 800GAUI-4, and *m*:8 PMA for 1.6TAUI-8, as specified in Clause 176.

The PMA below the 200 Gb/s per lane AUI-C2C/C2M is any 1:*n* PMA for 200GAUI-1, 2:*n* PMA for 400GAUI-2, 4:*n* PMA for 800GAUI-4, and 8:*n* PMA for 1.6TAUI-8, as specified in Clause 176.

The service interface above and below the 200 Gb/s per lane AUI-C2C/C2M is the PMA service interface as specified in 176.2.

The SIGNAL_OK parameter of the PMA:IS_SIGNAL.indication (for an AUI component above the AUI channel) or PMA:IS_SIGNAL.request (for an AUI component below the AUI channel) corresponds to the variable training_status of the ILT function, as defined in 176A.11.2.1. When SIGNAL_OK is either IN_PROGRESS or FAIL, the corresponding tx_symbol parameters on all lanes are unspecified.

[Also, include the inter-sublayer interface figure shown on an earlier slide.]

Assigning values to variables

The variables adjacent_remote_rts and adjacent_isl_ready are defined in Draft 1.1 as shown top right.

These are not sufficiently complete.

Change these definitions as shown in the bottom right.

176A.11.2.1 Variables

adjacent_remote_rts
Boolean variable that is set to the value of remote_rts on the other interface of the device.

adjacent_isl_ready
Boolean variable that is set to the value of isl_ready on the other interface of the device.

Change the variable definitions and add the note as follows:

adjacent_remote_rts
Boolean variable that indicates the value of remote_rts on the other interface of the device. It is set to true if the parameter SIGNAL_OK is OK and is otherwise set to false.

adjacent_isl_ready
Boolean variable that indicates the value of isl_ready on the other interface of the device. It is set to true if the parameter SIGNAL_OK is OK or READY and is otherwise set to false.

NOTE – SIGNAL_OK is received via the IS_SIGNAL.request primitive for an AUI component above an AUI channel or a PMD, or via the IS_SIGNAL.indication primitive for an AUI component below an AUI channel.

xBASE-R SM-PMA (Clause 176)

- To accommodate ILT, the intermediate PMA will need to do the follow:
  - Consider SIGNAL_OK states on each input.
    - Update the state diagram(s) appropriately.
  - Modify SIGNAL_OK treatment at each output.
    - Set the state value on a combination of input SIGNAL_OK values and state diagram state(s).
  - Pass through clock from input to output in each direction.

PMA figures

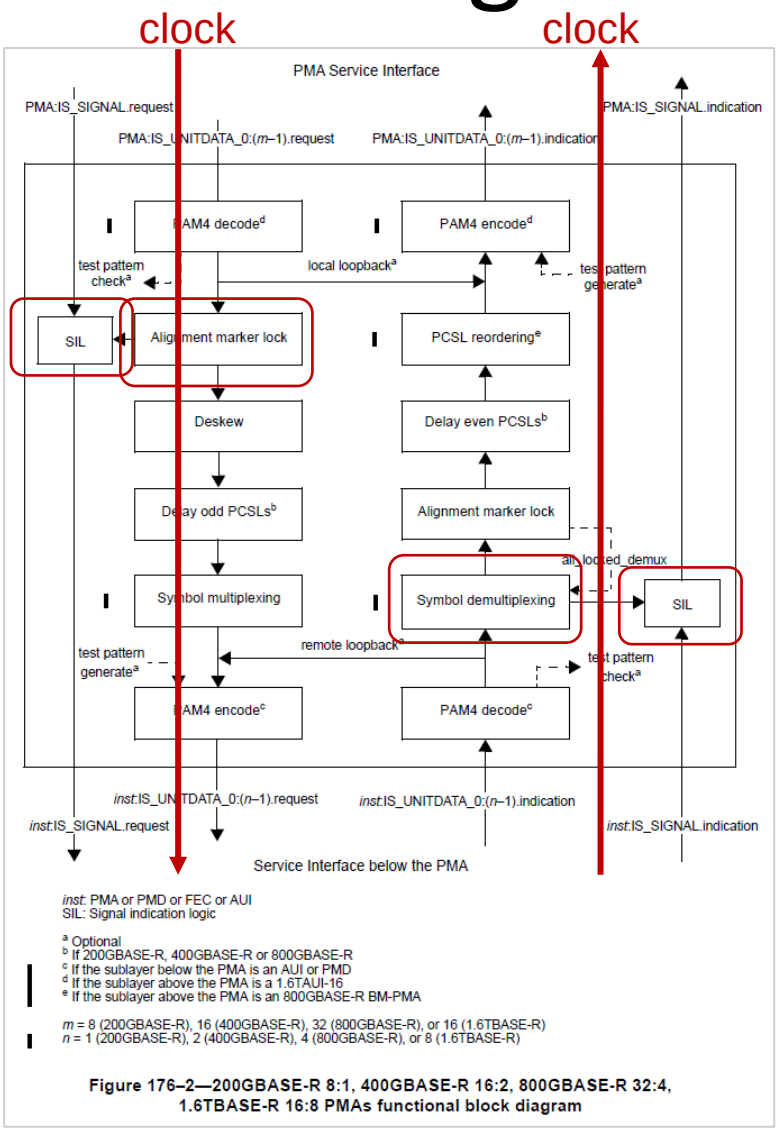


Figure 176–2—200GBASE-R 8:1, 400GBASE-R 16:2, 800GBASE-R 32:4, 1.6TBASE-R 16:8 PMAs functional block diagram

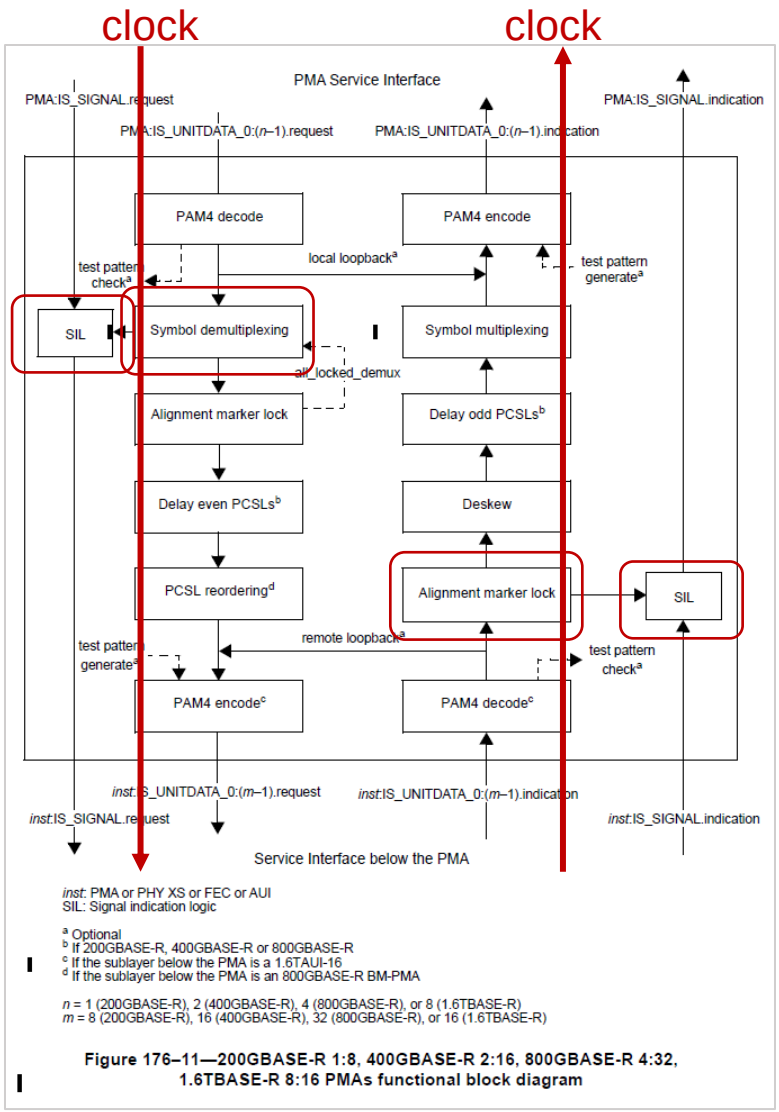


Figure 176–11—200GBASE-R 1:8, 400GBASE-R 2:16, 800GBASE-R 4:32, 1.6TBASE-R 8:16 PMAs functional block diagram

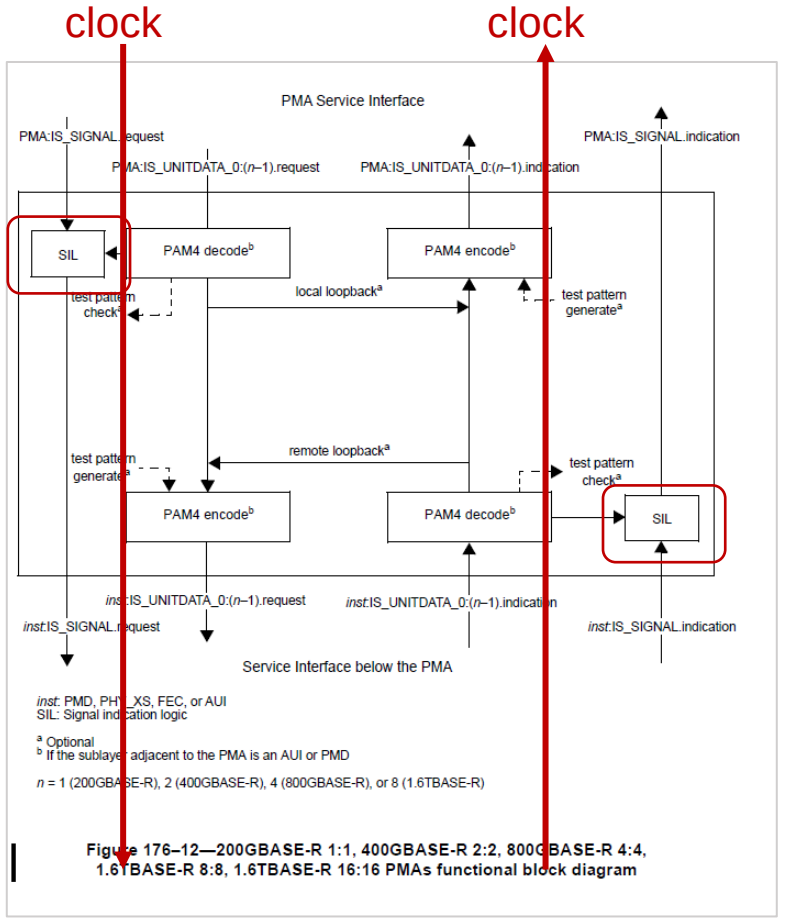


Figure 176–12—200GBASE-R 1:1, 400GBASE-R 2:2, 800GBASE-R 4:4, 1.6TBASE-R 8:8, 1.6TBASE-R 16:16 PMAs functional block diagram

This slide depicts where some consideration is required. This is not a proposal.

PMA state diagram variables

In 176.4.5.2.1 the signal_ok variable is set based on there being only two values for the parameter SIGNAL_OK (OK, FAIL). Since there are now four values (OK, IN_PROGRESS, READY, OK) the definition must be updated.

Also, it is confusing to define the same variable in each. Change the name in the variable list and diagrams.

Change the definitions for signal_ok (two instances) as follows:

signal_ok_mux

Boolean variable that is set based on the most recently received value of the SIGNAL_OK primitive on the service interface in the multiplexing direction. It is true if the value was OK and false ~~if the value was FAIL.~~ otherwise.

signal_ok_demux

Boolean variable that is set based on the most recently received value of the SIGNAL_OK primitive of the service interface in the demultiplexing direction. It is true if the value was OK and false ~~if the value was FAIL.~~ otherwise.

PMA SIGNAL_OK values at outputs

Add the following to (m:n and n:m PMA) 176.4.2.1, 176.4.2.2, 176.5.2.1, 176.5.2.2, adding the table only to 176.4.2.1.

The output parameter SIGNAL_OK is set according to Table 176-x.

Table 176-x – Output SIGNAL_OK values

Input SIGNAL_OK value	align_status_mux or align_status_demux	Output SIGNAL_OK
OK	true	OK
	false	READY
READY	N/A	READY
IN_PROGRESS	N/A	IN_PROGRESS
FAIL	N/A	FAIL

Add the following to 176.6.2.1 and 176.6.2.2 (n:n PMA) …

The output parameter SIGNAL_OK is set to the value of the received SIGNAL_OK value.

PMA clock

In 176.4, 176.5, and 176.5 add (with editorial license) text specifying that:

the clock is passed from the interface above to the interface below in the transmit direction

the clock is passed from the interface below to the interface above in the receive direction.

xBASE-R Inner FEC (Clause 177)

- Update SIGNAL_OK and state machine definitions in a similar way as proposed for Clause 176.
- Specify that the clock is passed from the PMD below to the PMA above and vice versa in a similar way as proposed for Clause 176.

Thanks