802.3dj D3.0 Comment Resolution Logic Topics

Eugene Opsasnick (Broadcom), Logic Track Co-Lead Editor
Gary Nicholl (Cisco), Logic Track Co-Lead Editor
Matt Brown (Qualcomm), 802.3dj Chief Editor
Arthur Marris (Independant), Logic Track Editor

Introduction

- This slide package was assembled by the 802.3dj editorial team to provide background and detailed resolutions to aid in comment resolution.
- Specifically, these slides are for the various logic-topic comments.

Stateless decoder

Comments #360

Stateless decoder Comment #360

Cl **172** *SC* **172.2.5** *P* **262** *L* **42** # 360

Slavick, Jeff — Broadcom Inc

Comment Type **TR** *Comment Status* **D** *Stateless decoder (L)*

We have editted the 64b66b encoder/decoder sections of 800G to guide folks to use the new stateless decoder defined in 119. In Clause 119 FEC decoder we state when using the 119 stateless decoding function you SHALL also error mark extra blocks that could have errors due to scrambler extension.
In 172.2.5.3 FEC decodoer we have the following:
The Reed-Solomon decoder is identical to that specified in 119.2.5.3, with the following exceptions:
ù FEC_degraded_SER in flow 0 and flow 1 are mapped to FEC_degraded_SER_0 and FEC_degraded_SER_1, respectively.
ù hi_ser in flow 0 and flow 1 are mapped to hi_ser_0 and hi_ser_1, respectively.

Which if someone is doing a 800G that uses the 119 stateless 64b66b decoder then the FEC decoder shall error mark some of the extra blocks beyond the uncorrectable CW. But if they're using the 172 stateless 64b66b decoder (perhaps due to re-use) then they would not be guided to fix their design to address the scrambler error extension possiblity. So adding guidance to do the same error mark when using 172 stateless 64b66b decoder, but not mandating it, would be a service to humanity to address a hole in the standard.

SuggestedRemedy

Bring in 172.2.5.3 add another exception that states:
"The additional error marking when using the stateless decoder specified in 119.2.5.8 should also be done if using the stateless decoder specified in 172.2.5.9.2."

Proposed Response *Response Status* **W**

PROPOSED REJECT.
The comment is essentially a repeat of comment #459 against 802.3dj D2.1 (https://www.ieee802.org/3/dj/comments/D2p1/8023dj_D2p1_comments_unsatisfied_id.pdf)

Initial proposed response is REJECT, mainly based on:

1. The old stateless 64B/66B decoder defined in 802.3df in CL 172 and RS-FEC decoder does not mark error blocks for the “full error extension” of the descrambler. But we cannot add it and make implementations based on 802.3df non-compliant.
2. New designs are encouraged to use the new stateless 64B/66B decoder and correct error marking is defined in 802.3dj modifications to CL 119.

But maybe we can do something….

Stateless decoder - new options in CL 172
Comment #360

- New stateless 64B/66B option in CL 172 (with recommendation):

Change the text of 172.2.5.9 as follows:

The receive PCS decodes 66-bit blocks to produce RXD<63:0> and RXC<7:0> for transmission to the 800GMII. One 800GMII transfer is decoded from each 66-bit block. The receive PCS shall use one of ~~the~~ two decoding methods ~~that are defined in 172.2.5.9.1 and 172.2.5.9.2~~, a state-diagram based method as defined in 172.2.5.9.1 or a stateless method. If using a stateless method, the stateless decoder defined in 119.2.5.8.2 should be used while the stateless decoder defined in 172.2.5.9.2 may be used.

Stateless decoders - old and new

- Old stateless decoder in CL172:

Table 172–4—PCS stateless decoder rules

reset	R_TYPE(rx_coded$_{i-1}$)[a]	R_TYPE(rx_coded$_i$)[b]	Resulting rx_raw
1	any block type	any block type	LBLOCK_R
0	any block type	E	EBLOCK_R
0	E	any block type	EBLOCK_R
0	any combination not listed above		DECODE(rx_coded$_i$)

[a] rx_coded$_{i-1}$ is the 66-bit block that immediately precedes rx_coded$_i$.
[b] rx_coded$_i$ is the 66-bit block that is being decoded.

- New stateless decoder in CL119:

119.2.5.8.2 Stateless decoder

The stateless decoder generates 200GMII/400GMII transfers based only on the current 66-bit block and PCS reset. When PCS reset is asserted, RXD<63:0> and RXC<7:0> are set to the constant LBLOCK_R (see 119.2.6.2.1). When PCS reset is not asserted, RXD<63:0> and RXC<7:0> are decoded from rx_coded<65:0> as defined in 119.2.3.

CL 172 stateless 64B/66B decoder extends error block marking by one 66-bit block.

CL 119 stateless 64B/66B decoder does no error extension (“truly stateless”). Error extension is done in RS-FEC decoder.

Stateless decoder - CL 119 RS-FEC decoder error extension

802.3dj, D2.3, updated the 3rd paragraph in **119.2.5.3 “Reed-Solomon decoder”:**

> If bypass error indication is not supported or not enabled, when the Reed-Solomon decoder determines that a codeword contains errors that were not corrected, it shall cause the PCS receive function to set every 66-bit block within the two associated codewords to an error block (set to EBLOCK_R). This may be achieved by setting the synchronization header to 11 for all 66-bit blocks created from these codewords by the 256B/257B to 64B/66B transcoder. When the stateless 64B/66B decoder defined in 119.2.5.8.2 is used, then the first four 66-bit blocks (accounting for any alignment marker removal) following the associated codewords shall also be set to error blocks.

172.2.5.3 “RS-FEC decoder” is defined as:

The Reed-Solomon decoder is identical to that specified in 119.2.5.3, with the following exceptions:

- **— FEC_degraded_SER in flow 0 and flow 1 are mapped to FEC_degraded_SER_0 and FEC_degraded_SER_1, respectively.**
- **— hi_ser in flow 0 and flow 1 are mapped to hi_ser_0 and hi_ser_1, respectively.**

Therefore, when the 800GE PCS uses the 119.2.5.8.2 stateless 64B/66B decoder option, it is also required do the “correct” error extension. But this is indirect, through the cross-reference and a little subtle.

Stateless decoder - New proposed text for 172.2.5.3 RS-FEC decoder:

For D3.1: Add the following text to the end of 172.2.5.3:

> If bypass error indication is not supported or not enabled, when the Reed-Solomon decoder determines that a codeword contains errors that were not corrected, it causes the PCS receive function to set every 66-bit block within the two associated codewords to an error block (set to EBLOCK_R). When the stateless 64B/66B decoder specified in 172.2.5.9.2 is used, it is recommended that the first four 66-bit blocks (accounting for any alignment marker removal) following the two associated uncorrected codewords are also set to error blocks. If the stateless 64B/66B decoder specified in 119.2.5.8.2 is used, the setting of the following four 66-bit blocks to error blocks is required.

This wording adds two key clarifications:

1. If the “old” 172 stateless 64B/66B decoder option is used, then correct error marking *should* be done, but is not required.
2. Points out that the correct error marking is required when using the “new” 119 stateless 64B/66B decoder, which did not appear as a requirement before in CL 172 – it was only inferred by the cross-reference and indirect requirement in 119.2.5.3.

AUI Management

Comments 365, [142, 363, 299, 318]

AUI Management - PMA and PMD register names

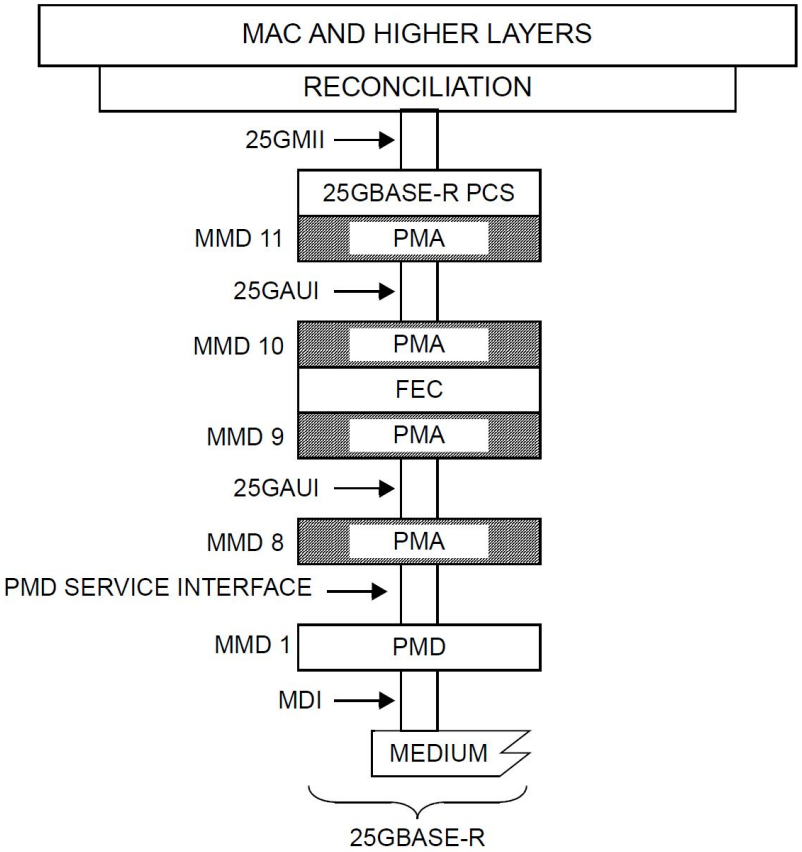
Table 45–1—MDIO Manageable Device addresses

Device address	MMD name
0	Reserved
1	PMA/PMD
2	WIS
3	PCS
4	PHY XS
5	DTE XS
6	TC

Table 45–1—MDIO Manageable Device addresses *(continued)*

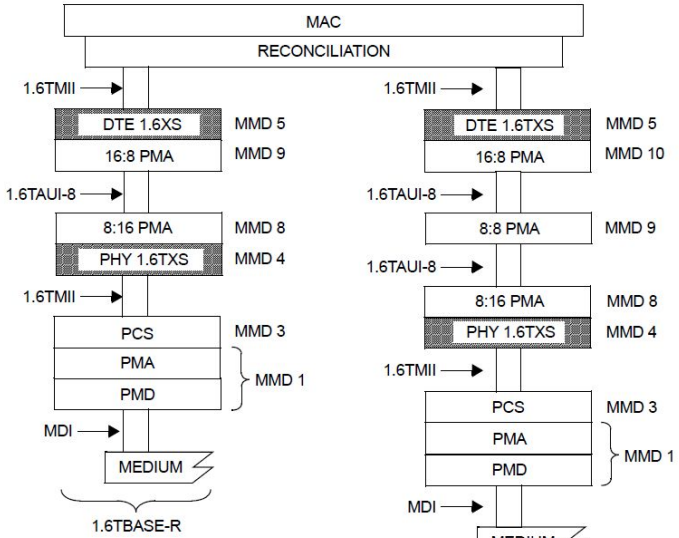
Device address	MMD name
7	Auto-Negotiation
8	Separated PMA (1)
9	Separated PMA (2)
10	Separated PMA (3)
11	Separated PMA (4)
12	OFDM PMA/PMD
13	Power Unit
14 through 28	Reserved
29	Clause 22 extension
30	Vendor specific 1
31	Vendor specific 2

MMD layering example (for 25G PHYs)



MMD layering example for 1.6T

Insert new Figure 171–3a after Figure 173–3 as follows:



1.6TAUI = 1.6 Tb/s ATTACHMENT UNIT INTERFACE
1.6TMII = 1.6 Tb/s MEDIA INDEPENDENT INTERFACE
1.6TXS = 1.6TMII EXTENDER SUBLAYER
DTE = DATA TERMINAL EQUIPMENT
MAC = MEDIA ACCESS CONTROL
MDI = MEDIUM DEPENDENT INTERFACE

MMD = MDIO MANAGEABLE DEVICE
PCS = PHYSICAL CODING SUBLAYER
PHY = PHYSICAL LAYER DEVICE
PMA = PHYSICAL MEDIUM ATTACHMENT
PMD = PHYSICAL MEDIUM DEPENDENT

Figure 171–3a—Example 1.6TBASE-R PMA layering with 1.6TXS

The MMD 1 address space is reused by the AUIs (see 45.2.1 in 802.3-2022)

From the base standard: "The addresses and functions of all registers in MMD 8, 9, 10, and 11 are defined identically to MMD 1"

45.2.1 PMA/PMD registers

For devices operating at 25 Gb/s or higher speeds, the PMA may be instantiated as multiple sublayers (see 83.1.4, 109.1.4, and 120.1.4 for how MMD addresses are allocated to multiple PMA sublayers for the respective speeds). A PMA sublayer that is packaged with the PMD is addressed as MMD 1. More addressable instances of PMA sublayers, each one separated from lower addressable instances, may be implemented and addressed as MMD 8, 9, 10, and 11 where MMD 8 is the closest to the PMD and MMD 11 is the furthest from the PMD. The addresses and functions of all registers in MMD 8, 9, 10, and 11 are defined identically to MMD 1, except registers m.5 and m.6 as defined in Table 45–2.

New in 802.3dj: AUI upper (host side) and AUI lower (network side)

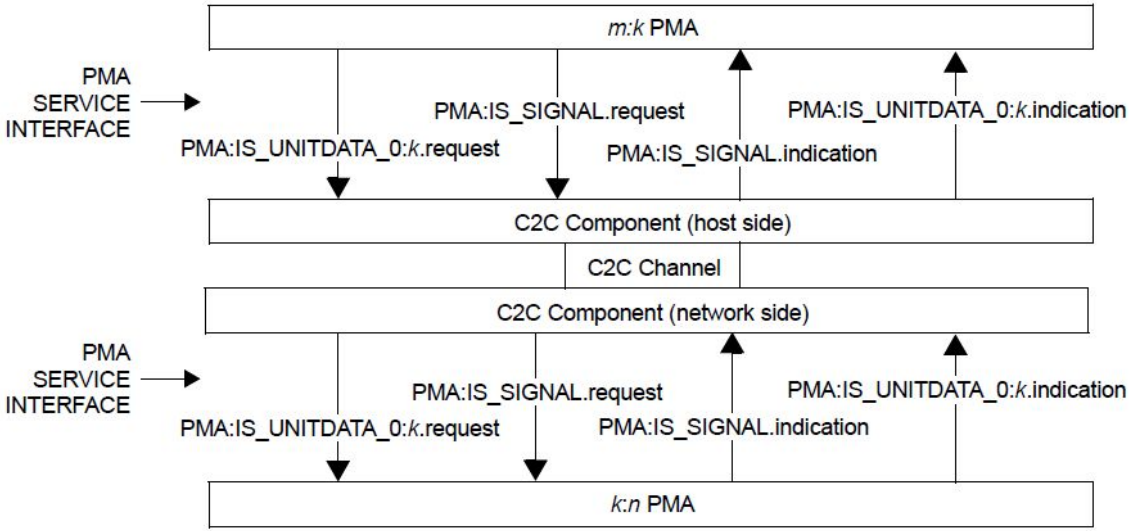


Figure 176C–3—Inter-sublayer service interfaces for a 200GAUI-1, 400GAUI-2, 800GAUI-4, or 1.6TAUI-8 C2C

AUI management

The base standard states in a sentence above Table 45-3 "The addresses and functions of all registers in MMD 8, 9, 10, and 11 are defined identically to MMD 1, except registers m.5 and m.6 as defined in Table 45–2", **therefore it is appropriate to reuse PMD registers for AUI/PMA control in other MMDs**.

Also it should be noted that in 802.3dj, the MMD 1 address space is duplicated for the upper (host side) AUI component as described in 45.2.1.272 so there are adequate control and status bits to cover the requirements of both the AUI and the PMD without aliasing to multiple variables.

45.2.1.272 Registers for the upper (host side) AUI component

The proposed response to comment #365 improves the text of 45.2.1.272 so that it will read as:

45.2.1.272 Registers for the upper (host side) AUI component (Registers 1.4000 through 1.7999)

Control and status registers are required for both the upper (host side) and lower (network side) AUI components (see Figure 176C-3 and Figure 176D-3). Registers 1.4000 through 1.7999 have identical functionality to the registers 1.0 through 1.3999 (address offset of 4000). The relevant registers from 1.0 through 1.3999 are used for control and status of the lower AUI component. The relevant registers from 1.4000 through 1.7999 are used for control and status of the upper AUI component. Register descriptions referring to the "PMD" should be interpreted as also referring to the “AUI”.

Comment #365:

CI **45** SC **45.2.1.272** P **123** L **25** # 365

Slavick, Jeff — Broadcom Inc

Comment Type **TR** *Comment Status* **D** *AUI Management (CG)*

The title of this section is not entirely true as it a replica of all the registers which is more than jus the ILT training registers.

SuggestedRemedy

Change the title of 45.2.1.272 to be "Registers for the upper AUI component"

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.

Change the title of 45.2.1.272 to be "Registers for the upper (host side) AUI component".

Change the first sentence from:
"Inter sublayer training requires control registers for the upper and lower AUI components."
To:
"Control and status registers are required for both the upper (host side) and lower (network side) AUI components (see Figure 176C-3 and Figure 176D-3)."

Add extra sentence to the end of the subclause:
"Register descriptions referring to the "PMD" should be interpreted as also referring to the AUI."

In Table 45-3 change:
"ILT training registers for the upper AUI component"
To:
"Registers for the upper (host side) AUI component."

Implement with editorial licence.

Comment #142 - part 1:

Cl **45** *SC* **45.2.1.8** *P* **82** *L* **1** # 142

Bruckman, Leon NVIDIA

Comment Type **TR** *Comment Status* **D** *AUI Management (CG)*

The title of this section is: "PMD transmit disable register", but in Table 45-3 of the base standard the name is: "PMA/PMD transmit disable". This makes the reference in 176C.3: "The C2C component shall meet the functional specifications in 178.8 and the management variable specifications in 178.13, unless stated otherwise." inconsistent since the name of the variables has the PMD prefix only (see comment #104 against D2.3)

SuggestedRemedy

Change the title of section 45.2.1.8 to: "PMA/PMD transmit disable register"

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.

The discrepancy (i.e. missing "PMA") between the subclause title "45.2.1.8 PMD transmit disable register" and the register name in Table 45-3 "PMA/PMD transmit disable" is in the base standard IEEE Std 802.3-2022. It is not within scope or necessary to change the subclause title as suggested.

The base standard states in a sentence above Table 45-3 "The addresses and functions of all registers in MMD 8, 9, 10, and 11 are defined identically to MMD 1, except registers m.5 and m.6 as defined in Table 45–2", therefore it is appropriate to reuse PMD registers for AUI/PMA control in other MMDs.

Also it should be noted that the MMD 1 address space is duplicated for the upper (host side) AUI component as described in 45.2.1.272 so there are adequate control and status bits to cover the requirements of both the AUI and the PMD without aliasing to multiple variables.

However the text in 176C.3 and 176D.3 should be clarified.

Comment #142 (*Proposed Response, continued*):

However the text in 176C.3 and 176D.3 should be clarified.

In 176C.3 on page 795 line 45 change from:
"The C2C component shall meet the functional specifications in 178.8 and the management variable specifications in 178.13, unless stated otherwise."
To:
"The C2C component shall meet the functional specifications in 178.8 unless stated otherwise. The C2C component shall use the management variable specifications in 179.14. The management variables in 179.14 have PMD in their variable names and point to MMD 1. The lower (network side) AUI component (see Figure 176C-3) will be located in a different MMD (8, 9, 10, or 11) and these PMD variables are reused for the AUI. For the upper (host side) AUI component the register addresses are offset by 4000 as described in 45.2.1.272."

In 176D.3 page 817 line 18, make the same change to "The C2M component shall meet the functional specifications in 179.8 and the management variable specifications in 179.14, unless stated otherwise."

AUI-C2M defined in 176D, AUI-C2C defined in 176C, and the KR and CR PMDs defined in 178 and 179 all use the same management variables as defined in 179.14.

Implement with editorial licence.
[CC 176C, 176D]

Comment #363:

Cl **179** SC **179.14** P **450** L **21** # 363

Slavick, Jeff — Broadcom Inc

Comment Type **T** *Comment Status* **D** *AUI management (CG)*

Retimers now have two PMA devices, and upper and lower. The 179.14 tables that C2M and C2C point to doesn't account for this (like is done in Table 178B-6).

SuggestedRemedy

Add the following footenote to Table 179-23 and 179-24 attached to the MDIO register/bit number heading
"MDIO register/bit numbers and references are for the lower AUI component or PMD. The MDIO register/bit numbers and references for the upper AUI component are at +4000 offset."

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.
AUIs are not the topic of clause 179. The proposed response to comment #142 addresses AUI MDIO registers with a change in Annexes 176C and 176D instead.
Resolve using the response to comment #142.

Comment #299:

Cl **176C** *SC* **176C.3** *P* **796** *L* **48** # 299

Dudek, Michael Marvell

Comment Type **TR** *Comment Status* **D** *AUI Management (CG)*

As discussed in dudek_3dj_01_2603 the C2C spec normatively requires meeting the functional specifications in management variable specifications in 178.13 which point to those in 179.14 where a number of registers are called PMD_xxxx (including PMD_reset). This is confusing particularly as PMA_reset is defined in table 178B-6

SuggestedRemedy

Adopt Option 1 on slide 6 of dudek_3dj_01_2603

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.
The referenced presentation is <https://www.ieee802.org/3/dj/public/26_03/dudek_3dj_01_2603.pdf>.

Resolve using the response to comment #142.

Comment #318 - part 1

Cl **176C** *SC* **176C.3** *P* **795** *L* **48** # 318

Nicholl, Gary — Cisco Systems, Inc.

Comment Type **T** *Comment Status* **D** *AUI Management (CG)*

As Mike Dudek pointed out in Draft 2.3 comment #104 (https://www.ieee802.org/3/dj/comments/D2p3/8023dj_D2p3_comments_final_clause.pdf), the C2C spec normatively requires meeting the functional specifications in management variable specifications in 178.13 which point to those in 179.14 where PMD_reset is required.
" The C2C component shall meet the functional specifications in 178.8 and the management variable specifications in 178.13, unless stated otherwise."

However the C2C component functions were separated from optional functions that resided in the PMA clause in earlier revisions of the specification. Mike was therefore proposig to reference the PMA status/control variables in Clause 176 rather than the PMD status/control control variables in Clause 178.

One issue with this approach is that the PMA and AUI compontent sublayers are co-located and likely using the same MMD (MDIO Manageable Device) address and therefore we can't use "PMA_reset" to independently reset the PMA sublayer and the AUI componenent subalyer as Mike proposed (see Figure 83C-7 as an example of how this was done in the past).

A better solution would to define a separate set of control and status variables for the AUI component (both C2C and C2M) sublayers and add the aprropriate MDIO registers in Clause 45.

Same issue applies in 176D.3 for the C2M AUI Component.

Comment #318 (*continued*)

SuggestedRemedy

Add a new sub-clause in 176C called "Management Variables" (similar to 178.13) and define the specific control and status variables that are required for the AUI C2C component. Add associated new MDIO registers in Clause 45.

An alternative proposal would be to continue to reference the management variable specifications in 178.13 and the same MDIO registers in Clause 45, but with the local control and status variables names of "AUI_xxx" replacing "PMD_xxx". With this approach we could use the same MMD address for both the PMA and the AUI Component, but still have unique control and status variables (and associated MDIO registers) for both sub-layers.

Make the same change in 176D for the AUI C2M Component.

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.

AUI-C2M defined in 176D, AUI-C2C defined in 176C, and the KR and CR PMDs defined in 178 and 179 all use the same management variables that are defined in 179.14.

Resolve using the response to comment #142

Syntax

Exception list and state diagram figure reference

Comments #74, 53

Single Item Exception List vs. Single paragraph Comment #74

Cl **176** *SC* **176.4.2.2** *P* **322** *L* **28** # I-74

Ran, Adee — Cisco Systems, Inc.

Comment Type **E** *Comment Status* **X**

(Comment resubmitted from WG ballot)
"with the following exception: <dashed list>"
A single exception does not require a list (there are many such exceptions in the draft without a list).

SuggestedRemedy

Change the first paragraph to the following text:
"The alignment marker lock process for each input PCSL operates in the same manner as the xBASE-R PCS alignment marker lock process (see 119.2.5.1, 172.2.5.1, and 175.2.5.1), except that the restart_lock variable (see Figure 119–12) takes the value of restart_lock_mux, which is defined in 176.4.4.2.1 and set by the PMA multiplex synchronization state diagram (Figure 176–10).

Delete the dashed list.

Comment's main point:

Remove the dashed list with only one exception item.

Editorial suggestion:

The proposed text will result in a paragraph with a long and complex sentence. The dashed list in the current draft makes it easier for the reader to focus on the exception. While it is true that some exceptions are described in the draft without a list, in this case, using a list makes the text easier to follow.

State Diagram Figure Reference
Comment #53

Current text with list:

176.4.2.2 Alignment marker lock

The alignment marker lock process for each input PCSL operates in the same manner as the xBASE-R PCS alignment marker lock process (see 119.2.5.1, 172.2.5.1, and 175.2.5.1) with the following exception:

— The restart_lock variable in Figure 119–12 takes the value of restart_lock_mux which is set in the state diagram shown in Figure 176–10.

The highlighted text is also the subject of comment #53 and should be changed to:

"in the PMA multiplex synchronization state diagram (Figure 176-10),"

Single Item Exception List vs. Text-only - Comparison
Comment #53 and #74

Modified for comment #53 (keeping list):

The alignment marker lock process for each input PCSL operates in the same manner as the xBASE-R PCS alignment marker lock process (see 119.2.5.1, 172.2.5.1, and 175.2.5.1) with the following exception:

— The restart_lock variable in Figure 119–12 takes the value of restart_lock_mux which is set in the PMA multiplex synchronization state diagram (Figure 176–10).

Modified for comment #74 (no list):

The alignment marker lock process for each input PCSL operates in the same manner as the xBASE-R PCS alignment marker lock process (see 119.2.5.1, 172.2.5.1, and 175.2.5.1), except that the restart_lock variable (see Figure 119–12) takes the value of restart_lock_mux, which is defined in 176.4.4.2.1 and set by the PMA multiplex synchronization state diagram (Figure 176–10).

Syntax - “Let x be …”

Comments #423

Syntax - Use of “Let <variable> be …” Comment #I-423

CI 175	SC 175.2.4.6.1	P291	L 10	# I-423

Dawe, Piers J G — NVIDIA

Comment Type **TR** *Comment Status* **X**

802.3 is a specification, not an example, lecture or tutorial. It is definitive and should be written in proper formal English. am_x<119:0> is not an example; it is *the* name for this thing. Other sections use better language, for example "The variable tx_am_sf is set as follows".

SuggestedRemedy

Change:
Let am_x<119:0> be the alignment marker for PCS lane x, x=0 to 15, where bit 0 is the first bit transmitted.
am_x<119:0> is constructed as shown in Figure 175-3 using the values in Table 175-4 for each PCS lane number x.
to:
The alignment marker for PCS lane x is am_x<119:0>, where x=0 to 15 and bit 0 is the first bit transmitted. It is constructed as shown in Figure 175-3 using the values in Table 175-4 for each PCS lane number x.
Scrub the draft for similar misuse of "let".

Main points of the comment:

- 802.3 spec should use “proper formal English”.
- am_x<119:0> is *the* name for a thing.
- “Let am_x<119:0> be …” is not desirable.
- 802.3dj D3.0 should eliminate the use of the word “let”.

Syntax - Use of “Let <variable> be …”
Comment #I-423

- "Let am_x<119:0> be the alignment marker for PCS lane x, x=0 to 15, where bit 0 is the first bit transmitted." is grammatically correct.
  - Breakdown of the sentence:
    - "Let am_x<119:0> be the alignment marker for PCS lane x,": Defines the variable and its name.
    - "x=0 to 15,": Defines the range of the index variable.
    - "where bit 0 is the first bit transmitted": Specifies the ordering of the bits, clarifying that it is a 120-bit vector transmitted sequentially.

- The following sentence: “am_x<119:0> is constructed as shown in Figure 175–3 using the values in Table 175–4 for each PCS lane number x." is also grammatically correct and follows the standard conventions for high-level technical specifications.
  - Breakdown of Correctness:
    - Passive Voice: "is constructed" is appropriate here because the focus is on the object (the alignment marker) rather than the person or system doing the building.
    - Prepositional Phrases: The use of "as shown in" and "using the values in" clearly links the text to its supporting visual data (Figure 175–3 and Table 175–4).
    - Variable Consistency: Referring to "each PCS lane number x" maintains consistency with the previous sentence defining x.

Syntax - Use of “Let <variable> be …”
Comment #I-423

- "Let <variable> be …” is the standard was to introduce and define variables before being used in formulas.
  - It is nearly universal in IEEE 802.3
  - “Let” is used 56 times in IEEE Std 802.3-2022
  - “Let” is used 14 times in 802.3dj D3.0
- Some examples:
  - Clause 91 (FEC):
    - "Let c be the smallest value of j such that tx_coded_c<0>=1."
    - "Let tx_payloads<(64j+63):64j> = tx_coded_j<65:2> for j=0 to 3"
    - "Let am_tx_x<65:0> be the alignment marker for PCS lane x, x=0 to 19, where bit 0 is the first bit transmitted."
    - "Let rx_payloads be a vector representing the payloads of the four 66-bit blocks."
    - "Let f_c<3:0> = rx_coded_c<5:2> be the scrambled first nibble (based on transmission order) of the block type field for rx_coded_c."
  - Clause 100 (PMD):
    - "... let X dB be the commanded average power of an equivalent 6 MHz channel for the second OFDM channel minus commanded average power of an equivalent 6 MHz channel for the first OFDM channel"
    - "Let p represent the pilot or PHY Link preamble symbol before transmit IDFT…”
    - "Let S_dB be the average power of the ideal QAM data subcarrier constellation (not including pilots) expressed in dB."
  - Clause 101 (RS, PCS, PMA for EPoC):
    - "Let the output of the linear feedback shift register be wk, ..."
    - "Let this extended sequence of length (N + DSNcp + DSNrp) be defined as: ..."
    - "Let Highestbitload be the highest bit loading among all active subcarriers ..."
    - "Let m = 1 and a binary bit is x."

Syntax - Use of “Let am_x<119:0> be …” Comment #I-423

- 802.3dj, D3.0, 175.2.4.6.1:

Let am_*x*<119:0> be the alignment marker for PCS lane x, x=0 to 15, where bit 0 is the first bit transmitted.

am_*x*<119:0> is constructed as shown in Figure 175–3 using the values in Table 175–4 for each PCS lane number *x*.

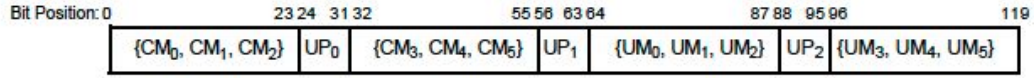


Figure 175–3—Alignment marker format

- IEEE Std 802.3-2022, 119.2.4.4.2:

Let am_*x*<119:0> be the alignment marker for PCS lane *x*, *x*=0 to 15, where bit 0 is the first bit transmitted. The alignment markers shall be mapped to am_mapped<1919:0> in a manner that yields the same result as the following process.

For *x*=0 to *15*, am_*x*<119:0> is constructed as follows:

am_*x*<119:0> is set to CM_0, CM_1, CM_2, UP_0, CM_3, CM_4, CM_5, UP_1, UM_0, UM_1, UM_2, UP_2, UM_3, UM_4 and UM_5, as shown in Figure 119–4 (bits 119:0) using the values in Table 119–2 for PCS lane number *x*.

The exact same sentence referenced in this comment in Clause 175 also exists in Clause 119 of the base standard.

It uses correct and proper English.

There is no question of misinterpretation.

Editorial suggestion:

There is no need to scrub the draft of “let”.
Let it be.