802.3dj D1.0 Comment Resolution Common Topics

Matt Brown (Alphawave Semi), 802.3dj Chief Editor
Leon Bruckman (Huawei)

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 common (not one specific track) comments.

Precoding

Precoding
Comments 21, 547, 582, 146, 145, 147, 148, 540, 541

Cl **176** *SC* **176** *P* **242** *L* **10** # 21

Liu, Cathy Broadcom

Comment Type **T** *Comment Status* **X**

In this section, precoding is mentioned to CR, KR and C2C links. How about C2M link? It should add C2M since C2M LT session specifies precoding as one of the options.

SuggestedRemedy

Add C2M link into the statement: "The precoding specifications in this subclause apply to the input and output lanes of a PMA that are connected to the service interface of an xBASE-CRn or xBASE-KRn PMD, or are part of an xAUI-n C2C/C2M link."

Proposed Response *Response Status* **O**

In addition to C2M links, all PAM4 optical PMDs require precoding as well. See next slide. Note that it is implicit that this PMA would not ever be connected to a PMD or AUI with lane signaling rates lower than 200 Gb/s, so no further clarification is required. [except 1.6TAUI-16]

Editor's recommendation:

Change the first sentence in 176.9.1.2 to the following:
"The precoding specifications in this subclause apply to the input and output lanes of a PMA that are connected to the service interface of an xBASE-KRn, xBASE-CRn, xBASE-DRn, or xBASE-FRn-500 PMD, or are part of an xAUI-n C2C or C2M link."

176.9.1.2 Precoding

The precoding specifications in this subclause apply to the input and output lanes of a PMA that are connected to the service interface of an xBASE-CRn or xBASE-KRn PMD, or are part of an xAUI-n C2C link.

The PMA shall provide 1/(1+D) mod 4 precoding capability on each transmit lane and may optionally provide 1/(1+D) mod 4 decoding capability on each receive lane. Precoding is implemented as specified in 135.5.7.2.

The precoder is enabled independently on the Tx output, Rx input, Rx output, and Tx input on each lane. Precoding is enabled and disabled using variables precoder_tx_out_enable_*i*, precoder_rx_in_enable_*i*, precoder_rx_out_enable_*i*, and precoder_tx_in_enable_*i* (where *i* is in the range 0 to 7).

If the PMA is connected to the service interface of an xBASE-CRn or xBASE-KRn PMD and training is enabled by the management variable mr_training_enable (see 136.7), then precoder_tx_out_enable_*i* and precoder_rx_in_enable_*i* shall be set as determined by the PMD control function in the LINK_READY state on lane *i* (see 136.8.11.7.5 and Figure 136–7). The method by which the PMD control function affects these variables is implementation dependent.

If the PMA is connected to the service interface of an xBASE-CRn or xBASE-KRn PMD and training is disabled by the management variable mr_training_enable, or if the PMA is part of an xAUI-n link, then precoder_tx_out_enable_*i*, precoder_rx_in_enable_*i*, precoder_rx_out_enable_*i*, and precoder_tx_in_enable_*i* are set as required by the implementation.

Precoding
Comments 21, 547, 582, 146, 145, 147, 148, 540, 541

Per motion #2 at the May Interim meeting, transmitter precoding was adopted along with OLT for all of the PMDs in Clause 180 through Clause 183.
For PMDs defined in Clause 180 and 181, the transmit precoding function is specified in Clause 176 (200G/400G/800G/1.6TBASE-R SM-PMA).
For PMDs defined in Clause 182 and 183, the transmit precoding function is specified in Clause 177 (200G/400G/800G/1.6TBASE-R Inner FEC)

Motion #2

Move to adopt OLT baseline per ghiasi_3dj_04a_2405 pages 3 and 4.

M: Ali Ghiasi
S: Matt Brown
Technical (>=75%)
802.3 voters only
Result: Y: 73 , N: 7, A: 13 Passed at 2:20 p.m.

14

https://www.ieee802.org/3/dj/public/24_05/motions_3dj_2405.pdf

Scope of OLT Proposal

- **Baseline proposal to to use CL176A AUI link training for optical link training OLT**
- **OLT baseline provides following function:**
  - Use the same training frame as CL176A structure
  - OLT baseline is based on CL176A but with some of fields changed to reserved
  - OLT uses relevant state diagram from CL176A (one not related to coefficient update)
    - For example, we don't need coefficient update state diagram
  - Propagating RTS (Ready to Send) status from PCS-AUI-optical-AUI-PCS
  - Precoder enable/disable
    - Transmitter pre-coder is mandatory to implement but optional to enable using OLT
- **Follow Control Function of Clause 179.8.9 PMD for optical clauses implementation**
  - Applies to all FECo and FECi relevant PMD clauses: CL 180, CL 181, CL 182, CL 183.

lighted

eiver is ready for data

Optics-LT – Ghiasi, et. al. IEEE 802.3dj Task Force 3

10	Reserved	Transmit as 0, ignore on receipt
9:7	Modulation and precoding request	9 8 7 1 1 1 = PAM4 free-running PRBS31 with precoding 1 0 1 = Reserved 1 1 0 = PAM4 free-running PRBS13 with precoding 1 0 0 = PAM4 PRBS13 0 1 1 = PAM4 free-running PRBS31 0 1 0 = PAM4 free-running PRBS13 0 0 1 = PAM2 free-running PRBS31 0 0 0 = PAM2 PRBS13
6:5	Reserved	Transmit as 0, ignore on receipt
4:2	Reserved	Transmit as 0, ignore on receipt
1:0	Reserved	Transmit as 0, ignore on receipt

		1 1 1 = PAM4 free-running PRBS31 with precoding 1 0 1 = Reserved 1 1 0 = PAM4 free-running PRBS13 with precoding 1 0 0 = PAM4 PRBS13 0 1 1 = PAM4 free-running PRBS31 0 1 0 = PAM4 free-running PRBS13 0 0 1 = PAM2 free-running PRBS31 0 0 0 = PAM2 PRBS13
9	Receiver frame lock	1 = Frame boundaries identified 0 = Frame boundaries not identified
8	Reserved	Transmit as 0, ignore on receipt
7	Parity	Even parity bit
6	Extended Training (RTS)	1 = No data is available, continue training 0 = Switch to data when training is completed
5:3	Reserved	Transmit as 0, ignore on receipt
2:0	Reserved	Transmit as 0, ignore on receipt

Optics-LT – Ghiasi, et. al. IEEE 802.3dj Task Force 4

https://www.ieee802.org/3/dj/public/24_05/ghiasi_3dj_04a_2405.pdf

Precoding
Comments 21, 547, 582, 146, 145, 147, 148, 540, 541

Cl **177** *SC* **177.4.7.2** *P* **256** *L* **13** # 582

Ghiasi, Ali Ghiasi Quantum/Marvell

Comment Type **T** *Comment Status* **X**

Pre-coding was shown on riani_3dj_01a_2303 FECI baseline that when was adopted, and pre-coding is essential for FECi PMDs

SuggestedRemedy

Please insert text for pre-coder in this sub-clause. as specified in 135.5.7.2, 120.5.7.2, and 173.5.7.2, 6 and 176.9.1.2, that may be enabled or disabled as needed with OLT, without OLT the optical transmitter should enable 1/(1+D) mod 4 precoding to mitigate burst error. See Ghiasi/Riani May-24 presentation on the need for pre-coder

Proposed Response *Response Status* **O**

Cl **177** *SC* **177.4.7.2** *P* **256** *L* **12** # 547

Rechtman, Zvi Nvidia

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

The 128,120 Hamming code is very sensitive to error propagation since it can correct up to one error in hard decoding and three errors in soft decoding. Hence, precoding is required

SuggestedRemedy

Add precoding, and use the same definition of precoding similar to 176.9.1.2.

Proposed Response *Response Status* **O**

177.4.7.1 Gray mapping

The Gray mapping for PAM4 encoded lanes is identical to that specified in 120.5.7.1.

177.4.7.2 Precoding

Editor's note (to be removed in the next draft):
Precoding was not explicitly called out in the adopted baselines. If necessary, a proposal is required. Otherwise this subclause will be removed.

Current specifications of precoding and gray mapping

120.5.7.1 Gray mapping for PAM4 encoded lanes

For output lanes encoded as PAM4 (for 200GBASE-R, where the number of output lanes is 4, or for 400GBASE-R, where the number of output lanes is 4 or 8), the PMA transmit process shall map consecutive pairs of bits {A, B}, where A is the bit arriving first, to a Gray-coded symbol as follows:

{0, 0} maps to 0,
{0, 1} maps to 1,
{1, 1} maps to 2, and
{1, 0} maps to 3.

For input lanes encoded as PAM4 (for 200GBASE-R, where the number of input lanes 4, or for 400GBASE-R, where the number of input lanes is 4 or 8), the PMA receive process shall map Gray-coded PAM4 symbols to pairs of bits {A, B} where A is considered to be the first bit as follows:

0 maps to {0, 0},
1 maps to {0, 1},
2 maps to {1, 1}, and
3 maps to {1, 0}.

Note that precoding and inverse precoding are not explicitly labelled. Rather they are defined as processes for input lanes and output lanes, respectively. Similar for Gray mapping and inverse Gray mapping.

When referencing these subclauses, we need to be careful with the language we choose.

135.5.7.2 Precoding for PAM4 encoded lanes

The precoding specifications in this subclause apply to the input and output lanes of a PMA that are connected to the service interface of a 50GBASE-R or 100GBASE-R PMD that includes the PMD control function defined in 136.8.11 (50GBASE-CR, 50GBASE-KR, 100GBASE-CR2, or 100GBASE-KR2), or are part of a 50GAUI-1 C2C or 100GAUI-2 C2C link.

The PMA shall provide 1/(1+D) mod 4 precoding capability on each output lane and may optionally provide 1/(1+D) mod 4 decoding capability on each input lane.

On each output lane, for each Gray-coded symbol $G(j)$, a precoded symbol $P(j)$ shall be determined by the following algorithm, where j is an index indicating the symbol number:

$P(j) = (G(j) - P(j-1)) \bmod 4$, when precoding is enabled (135–1)

$P(j) = G(j)$, when precoding is disabled (135–2)

On each input lane, for each precoded symbol $P(j)$, a Gray-code symbol $G(j)$ shall be determined by the following algorithm:

$G(j) = (P(j) + P(j-1)) \bmod 4$, when precoding is enabled (135–3)

$G(j) = P(j)$, when precoding is disabled (135–4)

The precoder is enabled independently for the input and output in each direction (Tx direction toward the PMD and Rx direction toward the MAC) and on each lane. Precoding is enabled and disabled using variables precoder_tx_out_enable_*i*, precoder_rx_in_enable_*i*, precoder_rx_out_enable_*i*, and precoder_tx_in_enable_*i*. If a Clause 45 MDIO is implemented, these variables are accessible through registers 1.600, 1.601, 1.602, and 1.603 (see 45.2.1.139 through 45.2.1.142). An example relating the variables with input and outputs is provided in Figure 135–7.

If the PMA is connected to the service interface of a PMD that includes the PMD control function and training is enabled by the management variable mr_training_enable (see 136.7), then precoder_tx_out_enable_*i* and precoder_rx_in_enable_*i* shall be set as determined by the PMD control function in the LINK_READY state on lane *i* (see 136.8.11.7.5 and Figure 136–7). The method by which the PMD control function affects these variables is implementation dependent.

If the PMA is connected to the service interface of a PMD that supports the PMD control function and training is disabled by the management variable mr_training_enable, or if the PMA is part of a 50GAUI-1 C2C or a 100GAUI-2 C2C link, then precoder_tx_out_enable_*i*, precoder_rx_in_enable_*i*, precoder_tx_in_enable_*i*, and precoder_rx_out_enable_*i* are set as required by the implementation. The method described in 135F.3.2.1 may be used for 50GAUI-1 C2C or 100GAUI-2 C2C.

Precoding
Comments 21, 547, 582, 146, 145, 147, 148, 540, 541

Editor's recommendation for 177.4.7

Replace 177.4.7 including the editor's note in 177.4.7.2 with the following:

177.4.7 PAM4 encoding
The PAM4 encoding function includes Gray mapping as specified in 177.4.7.1 and precoding as specified in 177.4.7.2.

177.4.7.1 Gray mapping
The Gray mapping for PAM4 encoded lanes is implemented as specified for output lanes in 120.5.7.1.

177.4.7.2 Precoding
The Inner FEC shall provide 1/(1+D) mod 4 precoding capability on each transmit lane.
Precoding is implemented as specified for output lanes in 135.5.7.2.
Tx precoding is enabled and disabled using variables precoder_tx_out_enable_i (where i is in the range 0 to 7).
If training is enabled by the management variable mr_training_enable (see 176A.11), then precoder_tx_out_enable_i shall be set as determined by the inter-sublayer link training function in the LINK_READY state on lane i (see Figure 176A-6). The method by which the inter-sublayer link training function affects this variable is implementation dependent.
If training is disabled by the management variable mr_training_enable precoder_tx_out_enable_i is set as required by the implementation (see 177.5.1).

Precoding
Comments 21, 547, 582, 146, 145, 147, 148, 540, 541

Editor's recommendation for 177.5…

Create new subclause prior to current 177.5.1 as follows:

177.5.1 PAM4 decoding
The PAM4 decoding function include inverse 1/(1+D) mod 4 precoding as specified in 177.5.1.1 and inverse gray mapping as specified in 177.5.1.2. Although the PAM4 decoding function is depicted as a discrete, serial function in Figure 177-2, it may be implemented anywhere in the receive function providing the net behaviour is the same.

177.5.1.1 Inverse precoding
The Inner FEC may optionally provide inverse 1/(1+D) mod 4 precoding capability on each receive lane.
If inverse precoding is implemented, it is enabled or disabled as determined by the implementation.
If inverse precoding is enabled, the Inner FEC receive function processes the detected data equivalent to the process specified for input lanes in 135.5.7.2.
If inter-sublayer link training function is enabled by the management variable mr_training_enable (see 176A.11), the precoding state on the link partner transmitter is requested using the inter-sublayer link training function.
If inter-sublayer link training function is disabled by the management variable mr_training_enable, the precoding state on the link partner transmitter is set by management.

177.5.1.2 Inverse Gray mapping
The inverse Gray mapping for PAM4 encoded lanes is identical to the process specified for input lanes in 120.5.7.1.

Precoding
Comments 21, 547, 582, 146, 145, 147, 148

Cl **181** *SC* **181.4** *P* **373** *L* **33** # 145

Ghiasi, Ali Ghiasi Quantum/Marvell

Comment Type **T** *Comment Status* **X**

Prior to 181.4 add section for PMA function to support precoder to mitigate burst errors

SuggestedRemedy

The transmitter need to supports 1/(1+D) mod 4 precoding, as specified in 135.5.7.2, 120.5.7.2, and 173.5.7.2, 6 and 176.9.1.2, that may be enabled or disabled as needed with OLT, without OLT the optical transmitter should enable 1/(1+D) mod 4 precoding to mitigate burst error.

Proposed Response *Response Status* **O**

Cl **180** *SC* **180.4** *P* **349** *L* **10** # 146

Ghiasi, Ali Ghiasi Quantum/Marvell

Comment Type **T** *Comment Status* **X**

Prior to 180.4 add section for PMA function to support precoder to mitigate burst errors

SuggestedRemedy

The transmitter need to supports 1/(1+D) mod 4 precoding, as specified in 135.5.7.2, 120.5.7.2, and 173.5.7.2, 6 and 176.9.1.2, that may be enabled or disabled as needed with OLT, without OLT the optical transmitter should enable 1/(1+D) mod 4 precoding to mitigate burst error.

Proposed Response *Response Status* **O**

Cl **182** *SC* **182.4** *P* **397** *L* **20** # 147

Ghiasi, Ali Ghiasi Quantum/Marvell

Comment Type **T** *Comment Status* **X**

Prior to 182.4 add section for PMA function to support precoder to mitigate burst errors

SuggestedRemedy

The transmitter need to supports 1/(1+D) mod 4 precoding, as specified in 135.5.7.2, 120.5.7.2, and 173.5.7.2, 6 and 176.9.1.2, that may be enabled or disabled as needed with OLT, without OLT the optical transmitter should enable 1/(1+D) mod 4 precoding to mitigate burst error.

Proposed Response *Response Status* **O**

Cl **183** *SC* **183.4** *P* **420** *L* **37** # 148

Ghiasi, Ali Ghiasi Quantum/Marvell

Comment Type **T** *Comment Status* **X**

Prior to 183.4 add section for PMA function to support precoder to mitigate burst errors

SuggestedRemedy

The transmitter need to supports 1/(1+D) mod 4 precoding, as specified in 135.5.7.2, 120.5.7.2, and 173.5.7.2, 6 and 176.9.1.2, that may be enabled or disabled as needed with OLT, without OLT the optical transmitter should enable 1/(1+D) mod 4 precoding to mitigate burst error.

Proposed Response *Response Status* **O**

These comments are resolved using the response provided on the previous page.

Training (Annex 176A)

Comment #575

Cl **176A** *SC* **176A.10.4** *P* **571** *L* **9** # 575

Law, David HPE

Comment Type **T** *Comment Status* **D** *ILT Diagrams*

The UPDATE_IC function is called in the OUT_OF_SYNC state of the Figure 176Aû9 Coefficient update state diagram. The UPDATE_IC function uses the ic_req variable to set the coefficients (see 176A.6.2), and the ic_req variable is derived from the 'initial condition request' bits from the control field of the received training frames (see 176A.10.3.1).

Since, however, the OUT_OF_SYNC state is entered during reset (reset or mr_restart set true), it would seem unlikely that training frames are being received. If that is the case, it isn't clear what the value of the ic_req variable is, and therefore what the coefficients should be set to.
á
176A.6.2 says that 'The transmitter equalizer is set to preset 1 upon entry to the QUIET state of the interface control state diagram.'. Since the QUIET state of the Interface control state diagram is also entered during reset, it seems the coefficients should be set to preset 1 when the Coefficient update state diagram is in the OUT_OF_SYNC state.

SuggestedRemedy

[1] Delete the first sentence of the ic_req definition in 176A.10.3.1.
[2] Add the text 'If the Coefficient update state diagram is in the OUT_OF_SYNC state ic_req is set to preset 1. Otherwise, it is derived from the "initial condition request" bit of the control field of received training frames on the correspondent lane of the interface.' to the end of the ic_req definition in 176A.10.3.1.

The comment is pointing out that the initial condition of the ic_req variable is not explicitly defined and there could be startup issue if PRESET1 is chosen over ind_sel as the default state.

Comment #575, Clause 176A

Rename and create a new ic_req

~~Ic_req~~ ic_sel

Enumerated variable derived from the “initial condition request” bits from the control field of the received training frames (see 176A.3). This variable may be assigned one of the following values (abbreviations used by the state diagram are included in parentheses): individual control (ind_ctl), preset 1, preset 2, preset 3, preset 4, preset 5.

ic_req

Variable that stores the most recent version of ic_sel.

Comment #575, Clause 176A

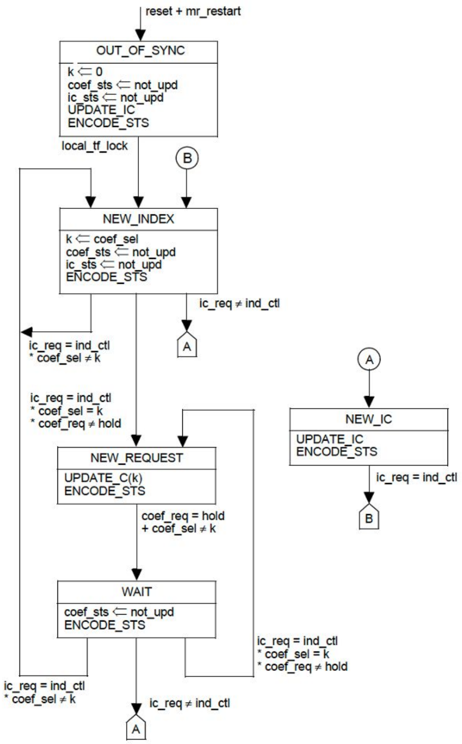


Figure 176A–9—Coefficient update state diagram

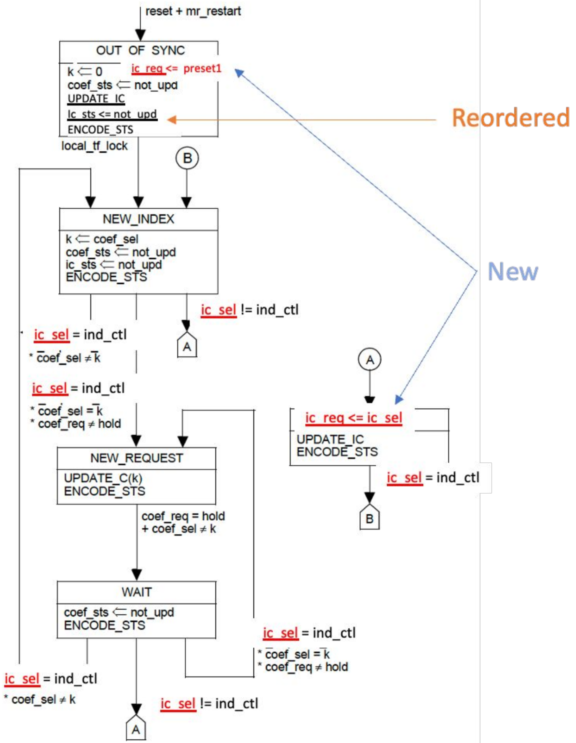


Figure 176A–9—Coefficient update state diagram

Training patterns

(comments #61, #200, #358, #496, #497, #498, and #548)

Comments #61, #200, #358, #496, #497, #498, and #548 are about the training patterns and are closely related.
We suggest choosing one of the options below.

Option 1: add more training pattern options

- Implement the suggested remedy in #358
  - The proposed changes would address #61, #200, #496, #497, #498 and #548
- Many changes to the draft

Option 2: keep the adopted training patterns, improve their definition

- Reject #61, #358, #497, and #498
- Implement the suggested remedies in #200 (which would address #496) and #548
- Fewer changes to the draft

Cl **176A** *SC* **176A.3** *P* **553** *L* **20** # 358

Healey, Adam Broadcom Inc.

Comment Type **T** *Comment Status* **X**

Training pattern options have been added to give receiver additional flexibility to successfully complete training. However, that flexibility is limited by a menu of fixed combinations of encoding and test pattern options. It would be better if encoding and test pattern selections were separated to allow receivers to request whatever combination best suits their needs. There is space in the control and status field structures to accommodate this.

SuggestedRemedy

In Table 176A-2, restore bits in control field bits 8 and 9 to the original "Modulation and precoding request" encoding defined in Clause 162. Define bits 5 and 6 to be "Test pattern request" with 00=PRBS13, 01=Free-running PRBS13, 10=Reserved, and 11=Free-running PRBS31. Restore bits 10 and 11 in the status field (Table 176A-3) to the "Modulation and precoding status" encoding defined in Clause 162. Define bits 12 and 13 to be "Test pattern status" using the same encodings as the control field. Update Figure 176A-2, 176A.3.2, and 176A.10.3.1 accordingly. Also add subclauses corresponding the Modulation and precoding request/status fields.

Proposed Response *Response Status* **O**

Table 176A-2 - Control field structure

Bit(s)	Name	Description
15:14	Reserved	Transmit as 0, ignore on receipt
13:11	Initial condition request	13 12 11 1 1 1 = Reserved 1 0 1 = Reserved 0 1 1 = Preset 5 0 0 1 = Preset 4 1 1 0 = Preset 3 1 0 0 = Preset 2 0 1 0 = Preset 1 0 0 0 = Individual coefficient control
10	Reserved	Transmit as 0, ignore on receipt
9:7	Pattern request	9 8 7 1 1 1 = PAM4 free-running PRBS31 with precoding 1 0 1 = Reserved 0 1 1 = PAM4 free-running PRBS31 0 0 1 = PAM2 free-running PRBS31 1 1 0 = PAM4 PRBS13 with precoding 1 0 0 = PAM4 PRBS13 0 1 0 = PAM4 free-running PRBS13 0 0 0 = PAM2 PRBS13
6:5	Reserved	Transmit as 0, ignore on receipt
4:2	Coefficient select	4 3 2 1 0 0 = Reserved 1 0 1 = $c(-3)$ 1 1 0 = $c(-2)$ 1 1 1 = $c(-1)$ 0 0 0 = $c(0)$ 0 0 1 = $c(1)$ 0 1 x = Reserved
1:0	Coefficient request	1 0 1 1 = No equalization 1 0 = Decrement 0 1 = Increment 0 0 = Hold

Option 1: add more training pattern options

- Implement the suggested remedy in #358
  - The proposed changes would address #61, #200, #496, #497, #498, and #548
- **This requires many changes to the draft**, as shown in the next 8 slides.

Modified Table 176A-2 - Control field structure

Bit(s)	Name	Description
15:14	Reserved	Transmit as 0, ignore on receipt
13:11	Initial condition request	13 12 11 1 1 1 = Reserved 1 0 1 = Reserved 0 1 1 = Preset 5 0 0 1 = Preset 4 1 1 0 = Preset 3 1 0 0 = Preset 2 0 1 0 = Preset 1 0 0 0 = Individual coefficient control
10	Reserved	Transmit as 0, ignore on receipt
9:8	Modulation and precoding request	9 8 1 1 = PAM4 with precoding 1 0 = PAM4 0 1 = Reserved 0 0 = PAM2
7:5	Reserved	Transmit as 0, ignore on receipt
4:2	Coefficient select	4 3 2 1 0 0 = Reserved 1 0 1 = *c*(−3) 1 1 0 = *c*(−2) 1 1 1 = *c*(−1) 0 0 0 = *c*(0) 0 0 1 = *c*(1) 0 1 x = Reserved
1:0	Coefficient request	1 0 1 1 = No equalization 1 0 = Decrement 0 1 = Increment 0 0 = Hold

Notes:

- Clause 162 compatibility: Set bits 6:5 = 00
- New training pattern options:
  - PAM2 free-running PRBS13
  - PAM4 free-running PRBS13 with precoding
- 4 reserved bits remaining in the control field
- 1 Modulation and precoding request reserved
- 1 Test pattern request reserved

7	Reserved	Transmit as 0, ignore on receipt
6:5	Test pattern request	6 5 0 0 PRBS13 0 1 free-running PRBS13 1 0 Reserved 1 1 free-running PRBS31

Modified Table 176A-3 - Status field structure

Bit(s)	Name	Description
15	Receiver ready	1 =Training is complete and the receiver is ready for data 0 =Request for training to continue
14:12	Reserved	Transmit as 0, ignore on receipt
11:10	Modulation and precoding status	11 10 1 1 = PAM4 with precoding 1 0 = PAM4 0 1 = Reserved 0 0 = PAM2
9	Receiver frame lock	1 = Frame boundaries identified 0 = Frame boundaries not identified
8	Initial condition status	1 = Updated 0 = Not updated
7	Parity	Even parity bit
6	Reserved	Transmit as 0, ignore on receipt
5:3	Coefficient select echo	5 4 3 1 0 1 = *c*(–3) 1 1 0 = *c*(–2) 1 1 1 = *c*(–1) 0 0 0 = *c*(0) 0 0 1 = *c*(1)
2:0	Coefficient status	2 1 0 1 1 1 = Reserved 1 1 0 = Coefficient at limit and equalization limit 1 0 1 = Reserved 1 0 0 = Equalization limit 0 1 1 = Coefficient not supported 0 1 0 = Coefficient at limit 0 0 1 = Updated 0 0 0 = Not updated

14	One	Transmit as 1
13:12	Test pattern status	13 12 0 0 PRBS13 0 1 free-running PRBS13 1 0 Reserved 1 1 free-running PRBS31

6	Extend training	1 = No data is available, continue training 0 = Switch to data when training is completed

Notes:
- No reserved bits remaining in the status field
- 1 Modulation and precoding status reserved
- 1 Test pattern status reserved

176A.2.3 Training pattern

The training pattern is the result of a training pattern generator equivalent to the structure shown in Figure 176A–2.

There are three PRBS generator functions: PRBS13, PRBS13 free-running, and PRBS31 free-running.

The training pattern selector is set to PAM2 PRBS13 upon entry to the QUIET state of the interface control state diagram (see Figure 176A–6).

NOTE—Exiting TRAINING mode requires both sides to use PAM4 modulation, thus the pattern request and the pattern status are both required to have values other than PAM2 PRBS13 and PAM2 free-running PRBS31.

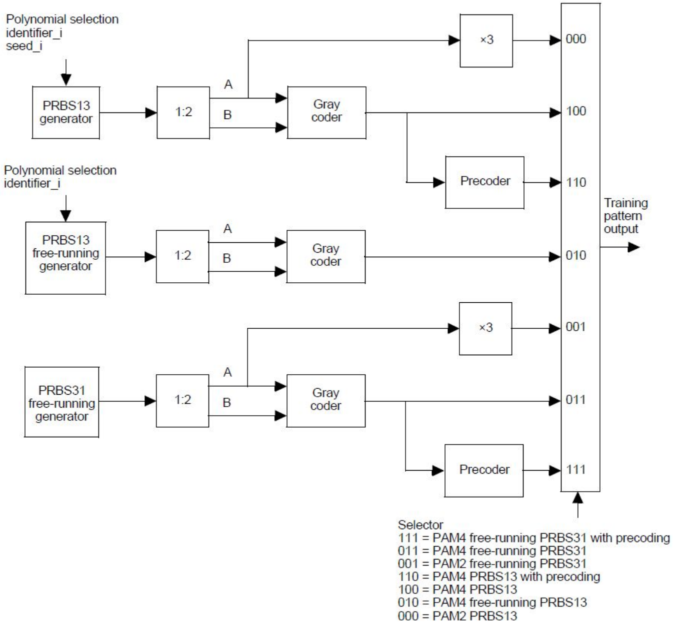


Figure 176A–2—Training pattern generator

176A.2.3 Training pattern

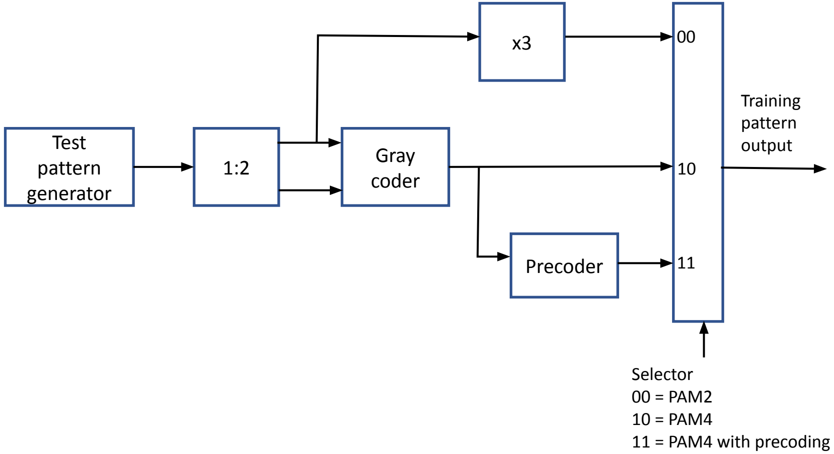
The training pattern is the result of a training pattern generator equivalent to the structure shown in Figure 176A–2.

There are three test pattern generator functions: PRBS13, PRBS13 free-running, and PRBS31 free-running.

The test pattern may be PAM2 or PAM4 modulated. For PAM4 modulation, precoding may be enabled or disabled.

The test pattern selector is set to PRBS13 and the modulation to PAM2 upon entry to the QUIET state of the interface control state diagram (see Figure 176A–6).

NOTE—Exiting TRAINING mode requires both sides to use PAM4 modulation, thus the modulation and precoding request and status are both required to have values other than PAM2.



176A.3.2 Pattern request

The pattern request bits are used to request that the link partner transmitter transmit one of the seven training pattern formats defined in 176A.2.3.

176A.x.x Modulation and precoding request

The modulation and precoding bits are used to request that the link partner transmitter modulates and codes the transmitted training pattern using one of the options defined in 176A.x.y (Test pattern modulation and coding).

176A.x.x Test pattern request

The test pattern bits are used to request that the link partner transmitter transmits one of the test patterns defined in 176A.2.3.1, 176A.2.3.2, and 176A.2.3.3.

176A.4.2 Pattern status

The pattern status bits encode the value of local_tp_mode.

176A.x.x Test pattern status

The test pattern status bits encode the value of local_tp_mode.

176A.x.x Modulation and precoding status

The modulation and precoding bits encode the value of local_md_mode.

176A.2.3.1 PRBS13 function

The PRBS13 function for each lane shall implement four generator polynomials. The polynomial used in each lane *i* is selected by the variable identifier_*i*.

At the start of the training pattern in each lane *i*, the state of the PRBS13 generator shall be set to the value of the variable seed_*i*. A value of all zeros is not valid.

Table 176A–1 specifies the default identifier, the corresponding polynomial, and the recommended default value of seed_*i* for each lane *i*, as well as the first 13 symbols of the training pattern for PAM2, PAM4, and PAM4 with precoding, created using the default polynomial and seed.

A sample implementation of the PRBS13 generator for identifier_*i* = 0 is shown in Figure 176A–3.

The PRBS13 generator can generate three different training patterns. The construction of each training pattern begins by demultiplexing the PRBS13 generator output into pairs of bits {A, B} where A corresponds to the 1st, 3rd, 5th, etc. bits output by the PRBS13 generator and B corresponds to the 2nd, 4th, 6th, etc. bits output by the PRBS13 generator. There is a new pair of bits each unit interval which implies that the PRBS13 generator is required to generate bits at twice the signaling rate. Given these pairs of bits, the three different training patterns correspond to three values of the training pattern selector in Figure 176A–2: PAM2 PRBS13, PAM4 PRBS13, and PAM4 PRBS13 with precoding.

When the training pattern selector is set to PAM2 PRBS13, the training pattern is the sequence of 16382 PAM4 symbols derived by mapping only the A bits such that logical 0 is transmitted as 0 and logical 1 is transmitted as 3.

When the training pattern selector is set to PAM4 PRBS13, the training pattern is the sequence of 16382 PAM4 symbols derived by Gray coding the {A, B} pairs as specified in 120.5.7.1.

When the training pattern selector is set to PAM4 with precoding, the training pattern is the sequence of 16 382 PAM4 symbols derived by Gray coding the {A, B} pairs as specified in 120.5.7.1 and precoding the result as specified in 135.5.7.2. The precoder state is initialized to 0 at the beginning of each training pattern, so that $P(j-1) = 0$ in Equation (135–1) for the first PAM4 symbol of the training pattern.

Two "0" symbols are transmitted immediately after the training pattern. This zero pad ensures the training frame is DC balanced and helps to delineate the start of the frame marker for the next training frame.

Move to new sub-clause: Test pattern modulation and coding

176A.2.3.2 PRBS13 free-running function

A set of four PRBS13 free-running generator polynomials are defined to minimize correlated interference between physical lanes, during the startup protocol. The PRBS13 free-running generator for each lane shall implement each of four generator polynomials defined in Table 176A–1. The polynomial used in each lane *i* is selectable by identifier_*i*. The default identifier for each lane is its lane number (e.g., the default value for identifier_0 is 0 which selects polynomial_0). A sample implementation of the PRBS13 free-running generator for identifier_*i* = 0 is shown in Figure 176A–3.

The initial value of the PRBS13 free-running generator shall not be all zeros. It may be any other value.

The construction of each training pattern begins by demultiplexing the PRBS13 free-running generator output into pairs of bits {A, B} where A corresponds to the 1st, 3rd, 5th, etc. bits output by the PRBS13 free-running generator and B corresponds to the 2nd, 4th, 6th, etc. bits output by the PRBS13 free-running generator. There is a new pair of bits each unit interval which implies that the PRBS13 free-running generator is required to generate bits at twice the signaling rate.

Move to new sub-clause: Test pattern modulation and coding

The PRBS13 free-running generator operates independently of the training frame and is not stopped or reset. As a result, the state of the PRBS13 free-running generator at the beginning of the training pattern changes between subsequent training frames. The marker, control, and status DME portions of the training frame (see 176A.2) periodically override the PRBS13 free-running generator output (for 288 UI every 16672 UI).

The training pattern is the sequence of symbols derived by Gray coding the {A, B} pairs as specified in 135.5.7.1.

Move to new sub-clause: Test pattern modulation and coding

176A.2.3.3 PRBS31 free-running function

The PRBS31 free-running generator for each lane shall implement the PRBS31 free-running pattern defined by Equation 49-2 and shown in Figure 49-9.

The PRBS31 free-running generator can generate three possible training patterns, PAM2 free-running PRBS31, PAM4 free-running PRBS31, and PAM4 free-running PRBS31 with precoding. These three variations are produced as described for the PRBS13 free-running function in 176A.2.3.2.

Move to new sub-clause: Test pattern modulation and coding

The initial state of the PRBS31 generator shall not be all zeros. It may be any other value. The PRBS31 generators in a multi-lane device shall be configured such that their relative offsets are large enough to make adjacent lanes uncorrelated within the length of the training frame. This can be achieved, for example, by initialization with different seeds or with the same seed at different times.

The PRBS31 free-running generator operates independently of the training frame and is not stopped or reset. As a result, the state of the PRBS31 free-running generator at the beginning of the training pattern changes between subsequent training frames. The marker, control, and status DME portions of the training frame (see 176A.2) periodically override the PRBS31 free-running generator output (for 288 UI every 16672 UI).

New sub-clause:

176A.x.x Test pattern modulation and coding

The training pattern generator can generate three different training patterns. The construction of each training pattern begins by demultiplexing the test pattern generator output into pairs of bits {A, B} where A corresponds to the 1st, 3rd, 5th, etc. bits output by the test pattern generator and B corresponds to the 2nd, 4th, 6th, etc. bits output by the test pattern generator. There is a new pair of bits each unit interval which implies that the test pattern generator is required to generate bits at twice the signaling rate. Given these pairs of bits, the three different training patterns correspond to three values of the training pattern selector in Figure 176A–2: PAM2, PAM4, and PAM4 with precoding.

When the training pattern selector is set to PAM2, the training pattern is the sequence of PAM4 symbols derived by mapping only the A bits such that logical 0 is transmitted as 0 and logical 1 is transmitted as 3.

When the training pattern selector is set to PAM4, the training pattern is the sequence of PAM4 symbols derived by Gray coding the {A, B} pairs as specified in 120.5.7.1.

When the training pattern selector is set to PAM4 with precoding, the training pattern is the sequence of PAM4 symbols derived by Gray coding the {A, B} pairs as specified in 120.5.7.1 and precoding the result as specified in 135.5.7.2.

For PRBS13, at the beginning of each training pattern the test pattern generator state is set to seed_*i* (see 176A.2.3.1) and the precoder state is set to 0 such that P(j–1) = 0 in Equation (135–1) for the first PAM4 symbol of the training pattern. For free-running PRBS13 and PRBS31, these operations are not performed.

Two “0” symbols are transmitted immediately after the training pattern. This zero pad delineates the start of the frame marker for the next training frame.

Cl **176A** *SC* **176A.2.3.3** *P* **552** *L* **46** # 498

Slavick, Jeff Broadcom

Comment Type **T** *Comment Status* **D** *ILT Pattern*

There is no zero pad for PRBS31 free-running. This means we could have a run length of 31 3's in a row when the maximal run length of the PRBS pattern runs into Frame Marker. The Zero pad is really part of the Framer Marker ensuring there is a distinct edge ahead of 16 UI run 3's for the start of the frame marker.

SuggestedRemedy

Bring the zero-pad back into the definition of the training frame. Stating that it is immediately precedes the training frame marker to provide a disticnt transition from training pattern to frame marker of the next training frame.

Proposed Response *Response Status* **W**

176A.7 Training pattern setting

When a change to the pattern request bits is detected, the transmitted training pattern (see 176A.2.3) is chosen accordingly, and the local_tp_mode variable is then set to the value of the pattern request bits to confirm that the change to the format of the training pattern was completed. local_tp_mode is encoded in the status field (see 176A.4.2).

176A.7 Training pattern setting

When a change to the modulation and precoding request bits or the test pattern request bits is detected, the transmitted training pattern (see 176A.2.3) is chosen accordingly. The local_mc_mode variable is then set to the value of the modulation and precoding request bits and the local_tp_mode variable to the value of the test pattern request bits, to confirm that the change to the format of the training pattern was completed. local_md_mode and local_tp_mode are encoded in the status field (see 176A.4.2).

local_tp_mode
Enumerated variable that controls the choice of the transmitted training pattern (see 176A.2.3) and may be assigned one of the following values: PAM2 PRBS13, PAM4 free-running PRBS13, PAM4 PRBS13, PAM4 PRBS13 with precoding, PAM2 free-running PRBS31, PAM4 free-running PRBS31, PAM4 free-running PRBS31 with precoding.

local_mc_mode
Enumerated variable that controls the choice of the transmitted training pattern (see 176A.2.3) modulation and coding, and may be assigned one of the following values: PAM2, PAM4, PAM4 with precoding.

local_tp_mode
Enumerated variable that controls the choice of the transmitted test pattern (see 176A.2.3), and may be assigned one of the following values: PRBS13, free-running PRBS13, free-running PRBS31.

Cl **176A** *SC* **176A.2.3.2** *P* **552** *L* **31** # 497

Slavick, Jeff Broadcom

Comment Type **T** *Comment Status* **X**

There is only 1 mode of operation for PRBS13 free-running, PAM4. We do have 1 free mode.

SuggestedRemedy

Add PRBS13-free running with precode as an option for a training pattern.

Proposed Response *Response Status* **O**

Included in new training pattern selection

Response:

ACCEPT IN PRINCIPLE. Resolve using the response to comment #358

Cl **176A** *SC* **176A.2.3.3** *P* **552** *L* **34** # 548

Rechtman, Zvi Nvidia

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

In the case of multi-lane operation, if all lanes exits the QUIET state simultaneously and use the same PRBS31 initial seed, there will be an undesired crosstalk effect. This potential issue needs to be addressed

SuggestedRemedy

Explicitly define that each lane must use different initial seed.

Proposed Response *Response Status* **O**

Included in new text of sub-clause

Response:

ACCEPT IN PRINCIPLE. Resolve using the response to comment #358

Cl **176A** *SC* **176A.2.3.3** *P* **552** *L* **40** # 200

Ran, Adee Cisco

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

"These three variations are produced as described for the PRBS13 free-running function in 176A.2.3.2"

PRBS13 free-running is defined only with PAM4 and does not have PAM2 or PAM4+precoding variants. These variants are defined for the PRBS13 function in 176A.2.3.1, but the definition of the precoding variant includes resetting of the precoder state at the beginning of each training frame, which would be inadequate.

SuggestedRemedy

Change to the following:

The initial state of the PRBS31 generator shall not be all zeros. It may be any other value.

When the training pattern selector is set to PAM4, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output.

When the training pattern selector is set to PAM2, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output, and the pair of bits {A, A} is used instead of {A, B}.

When the training pattern selector is set to PAM4 with precoding, the training pattern is generated from the PRBS31 PAM4 pattern by precoding the Gray-mapped PAM4 symbols as specified in 135.5.7.2. The precoder initial state is not specified. The state is not re-initialized or reset during generation of the training pattern.

Proposed Response *Response Status* **O**

Fixed in new text of sub-clauses

Response:

ACCEPT IN PRINCIPLE. Resolve using the response to comment #358

Cl **176A** *SC* **176A.4** *P* **555** *L* **17** # 61

Dudek, Mike Marvell

Comment Type **T** *Comment Status* **D** *ILT Frame*

It would be better to have the existing patterns the same as for previous clause 136.

SuggestedRemedy

In Table 176A-3 use the 1 in bit 12 for the new patterns keeping the bits 11 and 10 the same as they were in clause 136 i.e. change 010 to PAM4 PRBS13, 100 to PAM4 free running PRBS13, 011 to PAM4 PRB13 with precoding and 110 to PAM4 free-running PRBS31

Proposed Response *Response Status* **W**

Included in new training pattern selection

Response:

ACCEPT IN PRINCIPLE. Resolve using the response to comment #358

Option 2: keep the adopted training patterns (without changing or adding new ones), improve their definition

- Reject #61, #358, #497, and #498
- Implement the suggested remedies in #200 (which would address #496) and #548
- Fewer changes to the draft - shown in the next 3 slides

Cl **176A** *SC* **176A.2.3.3** *P* **552** *L* **40** # 200

Ran, Adee Cisco

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

"These three variations are produced as described for the PRBS13 free-running function in 176A.2.3.2"

PRBS13 free-running is defined only with PAM4 and does not have PAM2 or PAM4+precoding variants. These variants are defined for the PRBS13 function in 176A.2.3.1, but the definition of the precoding variant includes resetting of the precoder state at the beginning of each training frame, which would be inadequate.

SuggestedRemedy

Change to the following:

The initial state of the PRBS31 generator shall not be all zeros. It may be any other value.

When the training pattern selector is set to PAM4, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output.

When the training pattern selector is set to PAM2, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output, and the pair of bits {A, A} is used instead of {A, B}.

When the training pattern selector is set to PAM4 with precoding, the training pattern is generated from the PRBS31 PAM4 pattern by precoding the Gray-mapped PAM4 symbols as specified in 135.5.7.2. The precoder initial state is not specified. The state is not re-initialized or reset during generation of the training pattern.

Proposed Response *Response Status* **O**

176A.2.3.2 PRBS13 free-running function

A set of four PRBS13 free-running generator polynomials are defined to minimize correlated interference between physical lanes, during the startup protocol. The PRBS13 free-running generator for each lane shall implement each of four generator polynomials defined in Table 176A–1. The polynomial used in each lane *i* is selectable by identifier_*i*. The default identifier for each lane is its lane number (e.g., the default value for identifier_0 is 0 which selects polynomial_0). A sample implementation of the PRBS13 free-running generator for identifier_*i* = 0 is shown in Figure 176A–3.

The initial value of the PRBS13 free-running generator shall not be all zeros. It may be any other value.

The construction of each training pattern begins by demultiplexing the PRBS13 free-running generator output into pairs of bits {A, B} where A corresponds to the 1st, 3rd, 5th, etc. bits output by the PRBS13 free-running generator and B corresponds to the 2nd, 4th, 6th, etc. bits output by the PRBS13 free-running generator. There is a new pair of bits each unit interval which implies that the PRBS13 free-running generator is required to generate bits at twice the signaling rate.

The PRBS13 free-running generator operates independently of the training frame and is not stopped or reset. As a result, the state of the PRBS13 free-running generator at the beginning of the training pattern changes between subsequent training frames. The marker, control, and status DME portions of the training frame (see 176A.2) periodically override the PRBS13 free-running generator output (for 288 UI every 16672 UI).

The training pattern is the sequence of symbols derived by Gray coding the {A, B} pairs as specified in 135.5.7.1.

176A.2.3.3 PRBS31 free-running function

The PRBS31 free-running generator for each lane shall implement the PRBS31 free-running pattern defined by Equation 49-2 and shown in Figure 49-9.

The PRBS31 free-running generator can generate three possible training patterns, PAM2 free-running PRBS31, PAM4 free-running PRBS31, and PAM4 free-running PRBS31 with precoding. These three variations are produced as described for the PRBS13 free-running function in 176A.2.3.2.

The PRBS31 free-running generator operates independently of the training frame and is not stopped or reset. As a result, the state of the PRBS31 free-running generator at the beginning of the training pattern changes between subsequent training frames. The marker, control, and status DME portions of the training frame (see 176A.2) periodically override the PRBS31 free-running generator output (for 288 UI every 16672 UI).

Cl 176A *SC* 176A.2.3.2 *P* 552 *L* 31 # 497

Slavick, Jeff Broadcom

Comment Type T *Comment Status* X

There is only 1 mode of operation for PRBS13 free-running, PAM4. We do have 1 free mode.

SuggestedRemedy

Add PRBS13-free running with precode as an option for a training pattern.

Proposed Response *Response Status* O

Cl 176A *SC* 176A.2.3.3 *P* 552 *L* 34 # 548

Rechtman, Zvi Nvidia

Comment Type TR *Comment Status* X

In the case of multi-lane operation, if all lanes exits the QUIET state simultaneously and use the same PRBS31 initial seed, there will be an undesired crosstalk effect. This potential issue needs to be addressed

SuggestedRemedy

Explicitly define that each lane must use different initial seed.

Proposed Response *Response Status* O

Cl 176A *SC* 176A.2.3.3 *P* 552 *L* 40 # 200

Ran, Adee Cisco

Comment Type TR *Comment Status* X

"These three variations are produced as described for the PRBS13 free-running function in 176A.2.3.2"

PRBS13 free-running is defined only with PAM4 and does not have PAM2 or PAM4+precoding variants. These variants are defined for the PRBS13 function in 176A.2.3.1, but the definition of the precoding variant includes resetting of the precoder state at the beginning of each training frame, which would be inadequate.

SuggestedRemedy

Change to the following:

The initial state of the PRBS31 generator shall not be all zeros. It may be any other value.

When the training pattern selector is set to PAM4, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output.

When the training pattern selector is set to PAM2, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output, and the pair of bits {A, A} is used instead of {A, B}.

When the training pattern selector is set to PAM4 with precoding, the training pattern is generated from the PRBS31 PAM4 pattern by precoding the Gray-mapped PAM4 symbols as specified in 135.5.7.2. The precoder initial state is not specified. The state is not re-initialized or reset during generation of the training pattern.

Proposed Response *Response Status* O

ACCEPT IN PRINCIPLE.

If we accept comment #497 as suggested, then PRBS13 will include PRBS13 PAM4 with pre-coding, so only PAM2 is missing. Change: "These three variations are produced as described for the PRBS13 free-running function in 176A.2.3.2."

to: "The initial state of the PRBS31 generator shall not be all zeros. It may be any other value. The PRBS31 generators in a multi-lane device shall be configured such that their relative offsets are large enough to make adjacent lanes uncorrelated within the length of the training frame. This can be achieved, for example, by initialization with different seeds or with the same seed at different times.

When the training pattern selector is set to PAM4 or PAM4 with precoding, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output.

When the training pattern selector is set to PAM2, the training pattern is generated in a similar manner to the definition in 176A.2.3.2, except that PRBS31 generator output is used instead of PRBS13 generator output, and the pair of bits {A, A} is used instead of {A, B}."

Implement with editorial license.

Zero padding - Comment #498

Cl **176A** *SC* **176A.2.3.3** *P* **552** *L* **46** # 498

Slavick, Jeff Broadcom

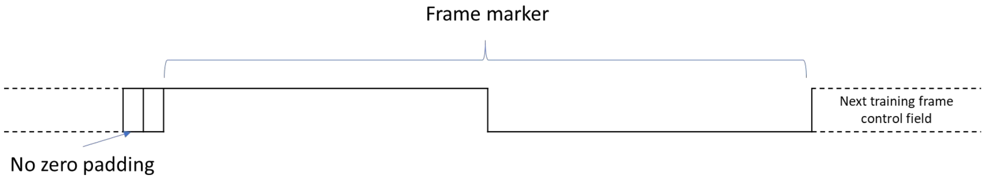
Comment Type **T** *Comment Status* **D** *ILT Pattern*

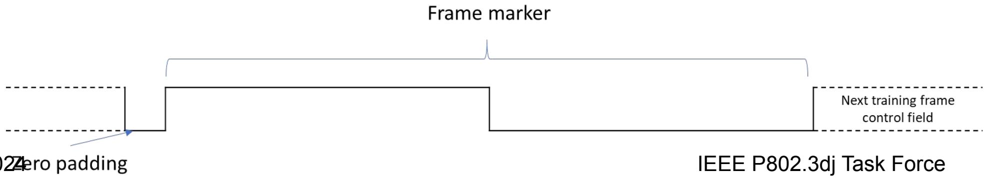
There is no zero pad for PRBS31 free-running. This means we could have a run length of 31 3's in a row when the maximal run length of the PRBS pattern runs into Frame Marker. The Zero pad is really part of the Framer Marker ensuring there is a distinct edge ahead of 16 UI run 3's for the start of the frame marker.

SuggestedRemedy

Bring the zero-pad back into the definition of the training frame. Stating that it is immediately precedes the training frame marker to provide a disticnt transition from training pattern to frame marker of the next training frame.

Proposed Response *Response Status* **W**





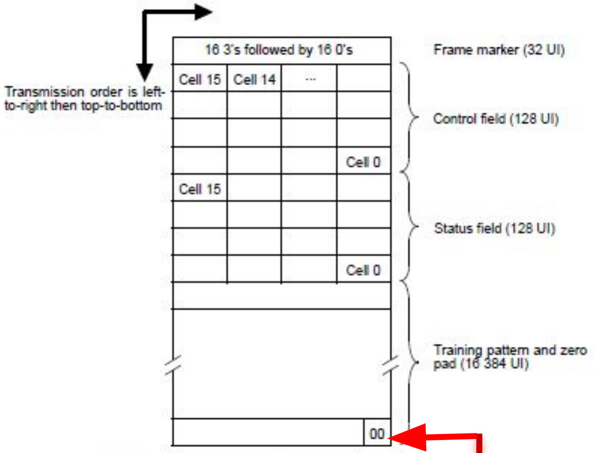


Figure 136–3—Training frame structure

These padd bits are not present in the training frame if the test pattern is free-running PRBS13 or PRBS31

Training
State diagrams

Comment #550

Cl **176A** *SC* **176A.10.4** *P* **568** *L* **48** # 550

Rechtman, Zvi Nvidia

Comment Type **T** *Comment Status* **D** *ILT Diagrams*

The comment refers to Figure 176Aû6ùInterface control state diagram.
The RECOVERY state coupled with the absence of timeouts, introduces a new challenge in identifying marginal performance cases. These cases may lead to repeated transitions between TRAIN_LOCAL/TRAIN_REMOTE/SEGMENT_READY state to/from RECOVERY state in scenarios of alternating local_tf_lock.
A possible solution is to limit the number of RECOVERY events by counting and limiting the number of transitions to the RECOVERY state.

SuggestedRemedy

Define a new counter: ôrecovery_event_countö. This counter increments each time the control state diagram transitions into the RECOVERY state.

Effects on the state diagram:
The ôrecovery_event_countö should be initialized to 0 in the ôSEND_TRAININGö state.
Upon entering the RECOVERY state, the ôrecovery_event_countö should be incremented by 1.

State diagram transition change:
The transition condition from the RECOVERY state to the FAIL state needs to be modified as follows:
Change ôrecovery_timer doneö to ôrecovery_timer done || recovery_event_count > Xö, where X is 5 (or to be determined).

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.

The suggested change has merit, but the suggested threshold of 5 is somewhat arbitrary. Depending on implementatation, other thresholds may be preferred, or this condition may be disabled, without affecting interoperability.

Implement the following with editorial license.

Define a new variable in 176A.10.3.1 as follows:
"max_recovery_events. Integer variable that controls the maximum allowed number of transitions into the RECOVERY state in the Interface control state diagram (Figure 176A-6). A value of zero allows unlimited number of transitions. The value of this variable is implementation dependent."

Define a new counter in 176A.10.3.4 as follows:
"recovery_event_count. This counter increments each time the control state diagram (see Figure 176A-6) transitions into the RECOVERY state."

In Figure 176A–6…
Initialize "recovery_event_count" to 0 in the "SEND_TRAINING" state.
In the RECOVERY state increment the "recovery_event_count" by 1.

Modify the transition condition from the RECOVERY state to the FAIL state as follows…
Change "recovery_timer done"
to "recovery_timer done + (max_recovery_events != 0)*(recovery_event_count >= max_recovery_events)".

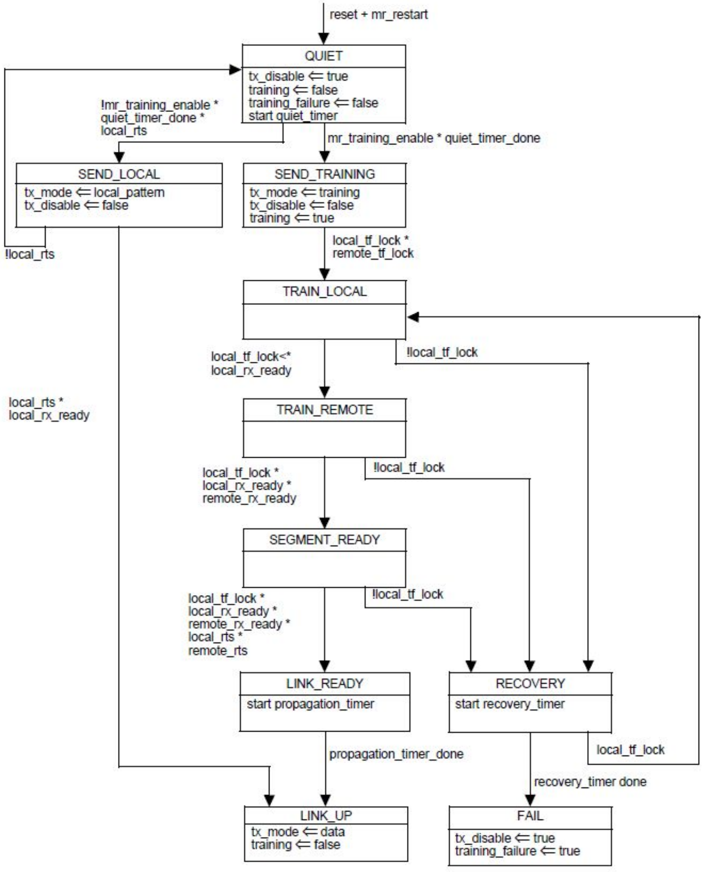


Figure 176A–6—Interface control state diagram

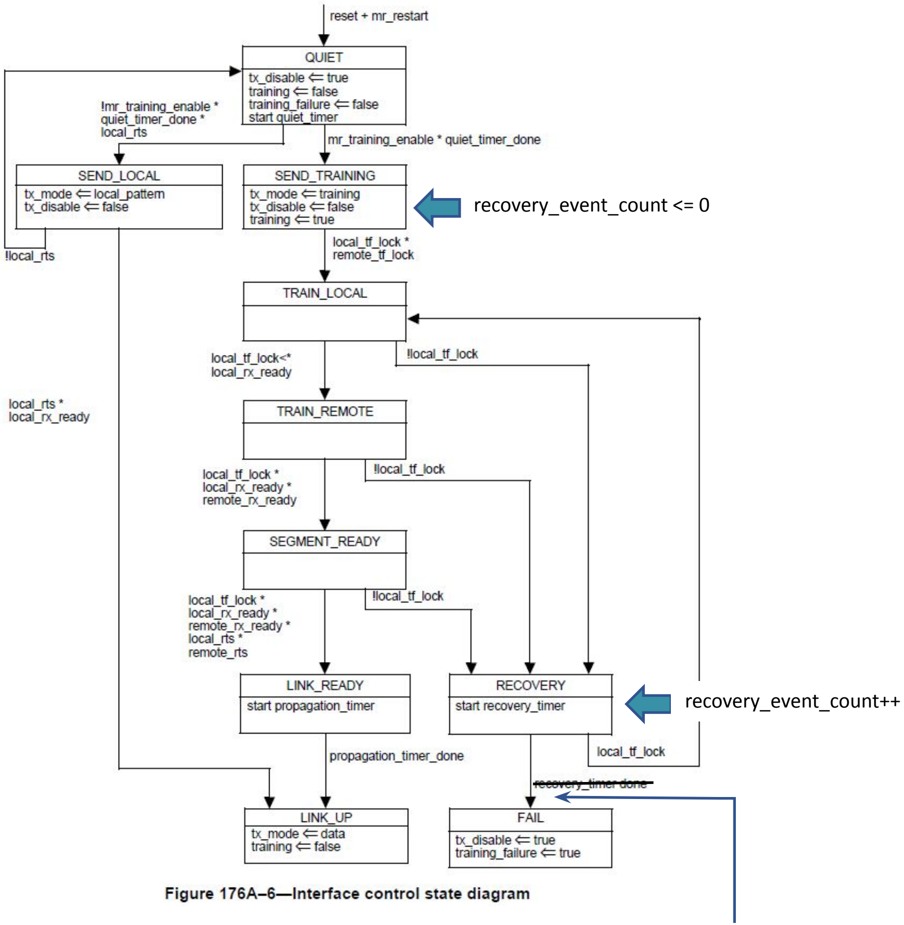


Figure 176A–6—Interface control state diagram

recovery_timer done +
(max_recovery_events != 0) * (recovery_event_count >=
max_recovery_events)