Note to Editor: Unchanged text shown in gray font (para/Figure numbering may have changed). Figures are thumbnails only. Notes such as this (titled "Note to Editor" are info only and not to be included in the draft).

142. Physical Coding Sublayer and Physical Media Attachment for Nx25G-EPON

This clause describes the Physical Coding Sublayer (PCS) with forward error correction (FEC) and Physical Medium Attachment (PMA) used with Nx25G-EPON point-to-multipoint (P2MP) networks. These are passive optical multipoint networks (PONs) that connect multiple DTEs using a single shared fiber. The architecture is asymmetric, based on a tree and branch topology utilizing passive optical splitters. This type of network requires that the Multipoint MAC Control sublayer exists above the MACs, as described in Clause 144.

In this clause the term xMII is used to refer to both the 25GMII and the XGMII interfaces.

142.1. Overview

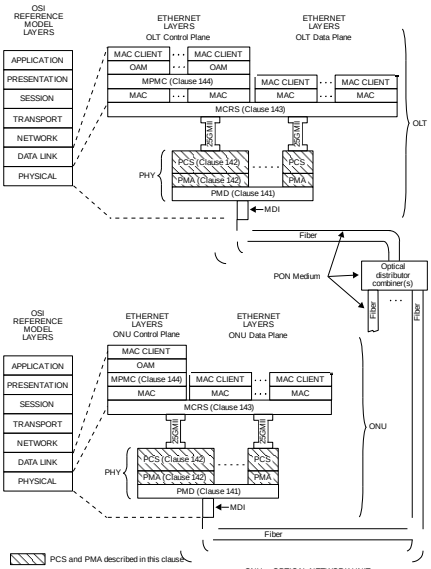

Figure 142–1—Relationship of EPON P2MP PMD to the ISO/IEC OSI reference model and the IEEE 802.3 Ethernet model

Figure 142-1-Relationship of EPON P2MP PMD to the ISO/IEC OSI reference model and the IEEE 802.3 Ethernet model

Note to Editor: the text below is from 142.2.1 Overview (with minor changes)

Figure 142-2 illustrates the functional block diagram of the Nx25G-EPON PHY with emphasis placed on the PCS. The Nx25G-EPON PCS is specified to support PQ-type PMDs, where:

— both the receive and transmit paths operate at 25.78125 Gb/s rate (25/25G-EPON, 50/25G-EPON, and 50/50G-EPON), or

— the receive path operates at 25.78125 Gb/s rate and the transmit path operates at 10.3125 Gb/s (25/10G-EPON and 50/10G-EPON ONU), or
— the transmit path operates at 25.78125 Gb/s rate and the receive path operates at 10.3125 Gb/s (25/10G-EPON and 50/10G-EPON OLT).

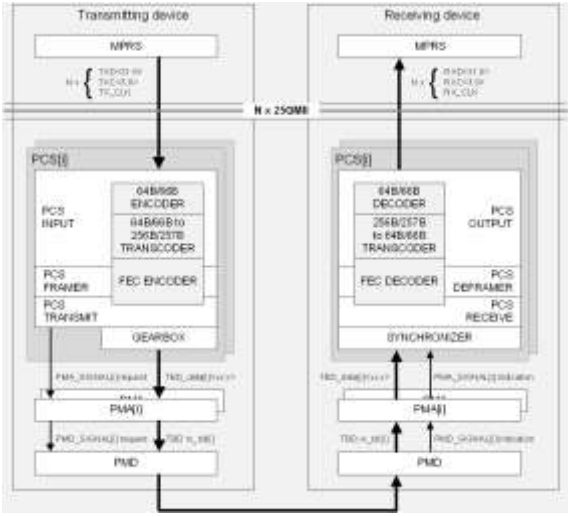

Figure 142-2 PCS Functional Block Diagram

142.1.1. Conventions

The notation used in the state diagrams in this clause follows the conventions in 21.5. Should there be a discrepancy between a state diagram and descriptive text, the state diagram prevails. The notation ++ after a variable indicates it is to be incremented by 1. The notation – – after a variable indicates it is to be decremented by 1. The notation –= after a variable indicates that the counter value is to be decremented by the following value. The notation += after a variable indicates that the counter value is to be incremented by the following value. Code examples given in this clause adhere to the style of the "C" programming language.

142.1.2. Delay constraints

{TBD}

142.1.3. Burst Transmission

Note to Editor: most of this section is taken from D1.2 142.2.2.2 PCS Framer.

Figure 142-3 and Figure 142-4 present the details of the ONU burst transmission, in particular, details of the FEC-unprotected and the FEC-protected portions of the upstream burst with several distinct Synchronization Pattern (SP) zones.

The upstream burst begins with a FEC-unprotected area comprising several explicit zones, each playing a separate role: SP_1 zone, optimized for laser on (T_{on}) and Automatic Gain Control (AGC, $T_{settling}$); SP_2 zone, optimized for Clock and Data Recovery (CDR, T_{CDR}); and SP_3 zone, optimized for the Start of Burst Delimiter (SBD) pattern. Each SP

Deleted: (Sync Pattern),

element is a multiple of 257 bits, aligning with the Clause 142 PCS line code of 256B/257B. This three zone arrangement is shown in Figure 142-3 for both normal operation (granted transmission) and the discovery operation. In the case of the discovery operation, the FEC-protected area comprises a single (shortened) FEC codeword.

Editor's Note (to be removed prior to publication): It was suggested that a better description of the content of the discovery operation burst might be needed, to make it clearer how many codewords and data blocks are present.

In some implementations, only two explicit zones are needed: SP_1 zone, optimized for laser on (T_{on}), Automatic Gain Control (AGC, $T_{settling}$), and Clock Data Recovery (CDR, T_{CDR}); and SP_2 zone, optimized for the Start of Burst Delimiter (SBD) pattern. This arrangement is shown in Figure 142-4 for the normal (granting operation) and discovery operation. In the case of the discovery operation, the FEC-protected area comprises a single (shortened) FEC codeword.

Bit patterns transmitted within each SP zone are configured using the SYNC_PATTERN MPCPDU (see <TBD new subclause with MPCPDU definition>) by the OLT which announces the number of the SP zones (two or three). The decision to use two or three SP zones is implementation-dependent and related to the design of the OLT burst-mode receiver.

In normal operation the SBD is followed by a number of FEC codewords, where the last codeword may be shortened to minimize the upstream transmission overhead from the long LDPC codewords used. Each FEC codeword comprises a series of 256B/257B encoded and scrambled data blocks, followed by a series of 257-bit long parity blocks. Within a non-shortened FEC codeword, the FEC payload portion includes 56 of these 257-bit data blocks and ten 257-bit parity blocks. Within a shortened FEC codeword, the FEC payload portion may be truncated to a number of data blocks smaller than 56, while the size of the FEC parity portion remains unchanged.

The upstream burst ends with an End of Burst Delimiter (EBD). When received at the OLT, the EBD pattern allows for the rapid reset of the OLT FEC synchronizer, preparing the OLT for the next incoming upstream burst. The EBD pattern is not part of the last FEC codeword.

The default number of Sync Pattern zones shall be 2, i.e., the *SP3Length* field value in the DISCOVERY MPCPDU and REGISTER MPCPDU is set to the value of zero (0).

The default configuration SP_1 Sync Pattern zone shall be defined as follows:

- covers T_{on}, $T_{setting}$, and T_{CDR};
- *SpValue*: 0x1-55-...-55
- *SpBalanced*: True (1)

The default configuration SP_2 Sync Pattern zone shall be defined as follows:

- covers SBD;
- *SpValue*: 0x1-BF-40-18-E5-C5-49-BB-59-6B-F8-D8-12-D8-58-E4-AB-40-BF-E7-1A-3A-B6-44-A6-94-07-27-ED-27-A7-1B-54
- *SpBalanced*: False (0)

Deleted: Sync Pattern

Deleted: Each of the Sync Pattern elements is a multiple of 257 bits, aligning with the Clause 142 PCS line code of 256B/257B.

Deleted: T

Deleted: 10

Deleted: of

Deleted: The OLT announces the number of the Sync Pattern zones (two or three) using the SYNC_PATTERN MPCPDU. The decision to use two or three Sync Pattern zones is implementation-dependent and related with the design of the OLT burst-mode receiver.¶

Deleted: are

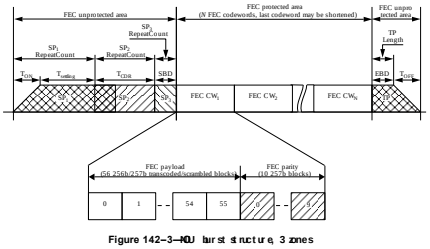

Figure 142–3—ONU burst structure, 3 zones

Figure 142-3 ONU burst structure, 3 zones

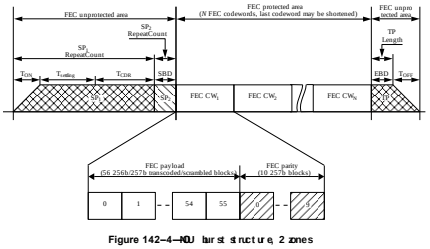

Figure 142–4—ONU burst structure, 2 zones

Figure 142-4 Burst structure, 2 zones

142.2. PCS transmit data path

Note to Editor: text from 142.2.2 PCS transmit function.

This subclause defines the transmit data path of the Nx25G-EPON PCS. In the OLT, the PCS transmit data path operates in a continuous mode at 25.78125 Gb/s rate. In the ONU, the PCS transmit data path operates in burst mode at 25.78125 Gb/s rate (25/25G-EPON, 50/25G-EPON, and 50/50G-EPON) or at 10.3125 Gb/s rate (25/10G-EPON and 50/10G-EPON). The PCS transmit data path includes a mandatory LDPC FEC encoder (see 142.2.4). The functional block diagram for the PCS transmit data path is shown in Figure 142-2 and consists of the following processes:

— PCS Input process (see **Error! Reference source not found.**),
— PCS Framer process (see **Error! Reference source not found.**)), and
— PCS Transmit process (see **Error! Reference source not found.**).

Note to Editor: additional text for this section will be in a separate comment.

142.2.1. 64B/66B line encoder

The Nx25G PCS encodes a 72-bit tx_raw vector into a 64B/66B block structure as defined in 49.2.4 with exceptions as noted in this subclause. The Nx25G-EPON PCS supports all the block type fields in Figure 49-7 except block type field values of: 0x2D, 0x33, 0x66, 0x55, and 0x4B. The Nx25G PCS uses the bit transmission order illustrated in Figure 142-5 rather than that shown in Figure 49-5.

Deleted: direction
Deleted: direction
Deleted: function
Deleted: direction
Deleted: direction
Deleted: functional blocks
Deleted: ,
Deleted:), and

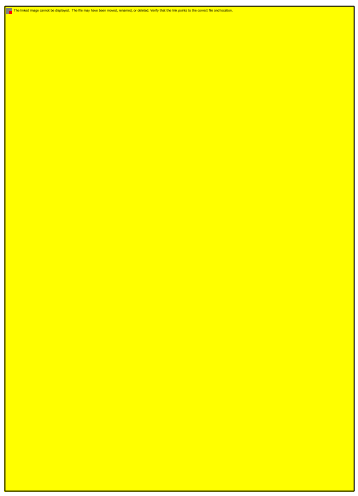

Figure 142-5 Transmit bit ordering

142.2.1.1. Control codes

Note to Editor: text from 142.2.2.1.2

The control characters and their mappings to Nx25G-EPON control codes are specified in Control Codes8023-142 proposed. The representations of the control characters are the control codes. Control characters are transferred over the xMII as 7-bit values. The Nx25G-EPON PCS encodes the start and terminate control characters implicitly using the block type field. The Nx25G-EPON PCS does not support ordered set control codes. All control code values that do not appear in Control Codes8023-142 proposed shall not be transmitted and are treated as an error if received.

Table 142–1—Control Codes8023-142 proposed

Control Character	Notation	xMII control code	Nx25GBASE-PQ control code
Idle	/I/	0x07	0x00
Inter Envelope Idle	/IEI/	0x08	0x08
Rate Adjust	/RA/	0x09	0x09
InterBurst Idle	/IBI/	0x09	0x09
Start	/S/	0xFB	Encoded by block type field
Terminate	/T/	0xFD	Encoded by block type field
Error	/E/	0xFE	0x1E

142.2.2. Scrambler

Prior to being transcoded into 257-bit blocks the Nx25G PCS scrambles four aggregated 66-bit blocks. The scrambling function is defined in 49.2.6.

142.2.3. 64B/66B to 256B/257B transcoder

Note to Editor: per 142.2.2.1.3

The 64B/66B to 256B/257B transcoder converts four consecutive scrambled 64B/66B blocks into one 256B/257B block as described in 91.5.2.5.

Deleted: scrambled

142.2.4. FEC encoder

Note to Editor: per D1.2 section 142.2.2.4

142.2.4.1. Low Density Parity Check coding

Note to Editor: per D1.2 section 142.2.2.4.1

142.2.4.2. LDPC encoder

Note to Editor: per D1.2 section 142.2.2.4.2

142.2.4.3. Encoding operation

Note to Editor: per D1.2 section 142.2.2.4.3

142.2.4.4. Transmit Interleaving

Note to Editor: per section 142.2.2.4.5.

142.2.4.5. Example of initial control seed sequence

Note to Editor: per D1.2 section 142.2.2.4.6

142.2.5. Transmit data path state diagrams

Various variables and buffers in the PCS are structured as 258-bit wide vectors with bits 0 through 256 holding one line-coding unit (a 257-bit block) and bit 257 conveying the origin of the block to be either the PCS Input process (bit 257 is one) or the PCS Framer process (bit 257 is zero). The value of bit 257 being one implies that the 257-bit block has been transcoded and scrambled.

Note to Editor: 142.2.5.1 to 142.2.5.3 are copied from D1.2 142.2.2.5.1 to 142.2.2.5.3 with some reordering. Any changes will be noted in separate comments.

142.2.5.1. Constants

EBD

TYPE: 258-bit vector
Value: {msb = 0, the remaining bits TBD}
The EBD constant holds the value of the end ofburst delimiter.

FEC_CW_DELIM

TYPE: 10-bit integer
Value: 0x3-CA
The constant is used for codeword alignment synchronization.

FEC_PARITY_SIZE

TYPE: Integer
Value: {10 TBC}

The FEC_PARITY_SIZE constant indicates the size of the Parity portion of a FEC codeword in units of 257 bits.

FEC_PAYLOAD_SIZE
TYPE: Integer
Value: {56 TBC}
The FEC_PAYLOAD_SIZE constant indicates the size of the Payload portion of a FEC codeword in units of 257 bits.

IBI
TYPE: 258-bit vector
Value: {msb = 0 the remaining bits TBD }
The IBI constant holds the value of the inter burst idle pattern.

IBI_EQ
TYPE: 72-bit vector
Value: 0xFF-0A-0A-0A-0A-0A-0A-0A-0A
The IBI_EQ constant indicates to the PCS that the burst is terminated.

PAR_PLACEHLDR
TYPE: 258-bit vector
Value: {msb = 0 the remaining bits TBD}
The PAR_PLACEHLDR constant represents the value of a 258-bit vector inserted into the data stream by the PCS Framer Process in order to reserve the location where FEC Parity and the FEC Delimiter is to be inserted into the data stream by the PCS Transmit Process.

RATE_ADJ_EQ
TYPE 72-bit vector
Value: 0xFF-09-09-09-09-09-09-09-09
The value of an EQ which is used as a placeholder to allow for rate differences between the MAC and the PHY layers.

142.2.5.2. Variables

CLK_IN
TYPE: Boolean
The clear on read variable CLK_IN is set to true on each falling edge of the xMII clock.

CLK_OUT
TYPE: Boolean
The clear on read variable CLK_OUT is set to true once for each 257-bits of data output by the PMD.

CLK_XFR
TYPE: Boolean
The clear on read variable CLK_XFR is set to true once for each 257-bits of data output by the PMD.

INPUT_FIFO[]
TYPE: array of 258-bit vectors
The INPUT_FIFO receives data from the MCRS Input process and hands it off to the Framer process. Its primary function is to absorb data while the PCS is transmitting burst overhead or FEC Parity. This FIFO holds at most SP_COUNT elements.

PARITY_STAGING_BUFFER[]
TYPE: vector of 2570 bits

This variable holds the 2560-bit calculated parity value along with the 10-bit FEC_CW_DELIM value (see **Error! Reference source not found.**). The total size of 2570 bits represents the same size as ten 257-bit ine encoding blocks.

ParityLeft

TYPE: Integer

The ParityLeft variable indicates the number of 257-bit Parity vectors needed to complete the current FEC codeword being processed by the PCS Framer.

PayloadLeft

TYPE: Integer

The PayloadLeft variable indicates the number of 257-bit payload vectors needed to complete the current FEC codeword being processed by the PCS Framer.

SP[]

TYPE: array of 258-bit vectors

An array of SP_LENGTH elements. Each element consists of MSB 0 and the 257-bit blocks as provisioned for SP1, SP2, SP3 by the MPCP.

SP_LENGTH

TYPE: integer

The SP_LENGTH variable represents the length of the synchronization pattern as determined by the most recent settings of SP1_RepeatCount, SP2_RepeatCount and SP3_RepeatCount received at the MPMC layer.

SpIndex

TYPE: integer

The SpIndex variable is a pointer into the SP[] array that indicates which 258-bit vector from the array should be sent to the TX_FIFO.

Transmitting

TYPE: Boolean

The Transmitting variable indicates whether the device is transmitting or not.

TX_FIFO[]

TYPE: array of 258-bit vectors

The TX_FIFO[] holds information queued by the Framer process for output by the PCS and enforces a fixed delay that is implementation dependent. The fixed delay ensures the PHY has sufficient time to generate FEC Parity given that the Framer process inserts SP_LENGTH 257-bit blocks at the beginning of the burst. The length of the TX_FIFO[] is defined as: MAX{ FEC_DELAY - SP_LENGTH, 2 }

TxInput

TYPE: 258-bit vector

This variable holds one transcoded 257-bit vector prepended with a binary one indicating the 257-bit block originated in the MCRS Input process.

TxNext

TYPE: 72-bit vector

The next 72-bit vector to be processed by the MCRS Input process.

TxLast

TYPE: 258-bit vector

This variable holds the 258-bit that was read from TX_FIFO in the previous CLK_OUT cycle.

TxOutput
TYPE: 258-bit vector
This variable holds one 258-bit vector retrieved from the OUTPUT_FIFO

TxPrev
TYPE: 72-bit vector
This variable holds one 72-bit vector received by the PCS Input Process from the 25GMII in the previous CLK_IN cycle.

XBUFFER[]
TYPE: array of 66-bit vectors
This buffer holds four 66-bit vectors of 64B/66B encoded data to be transcoded into one 257-bit block.

xIndex
TYPE: Integer
An index into the XBUFFER indicating the number of encoded vectors contained in the buffer that are ready to be transcoded.

142.2.5.3. Functions

ENCODE(v)
This function performs 64B/66B encoding of a 72-bit vector v per 49.2.13.2.3 and returns the result.

FecParity()
Upon initiation, the first call to this function returns a vector containing the first 257 bits from the PARITY_STAGING_BUFFER, i.e. PARITY_STAGING_BUFFER<256:0>. Each subsequent call returns the subsequent 257 bits from the buffer. On the 10th call, the last 257 bits are returned, i.e., PARITY_STAGING_BUFFER<2569:2312>, and the function resets to return PARITY_STAGING_BUFFER<256:0> on the next call. This emulates a circular buffer of size 10 x 257-bits.

FEC_Encode(v)
This function passes a 257-bit vector v to the FEC engine for encoding.

FIFO.Append(v)
This function adds the vector v to the input of FIFO buffer.

FIFO.Fill(v)
This function writes vector v, to each element of FIFO buffer.

FIFO.GetHead()
This function returns the oldest (head) element in the FIFO buffer, and removes that element from the FIFO, decreasing the length by one.

FIFO.IsEmpty()
This function returns true if the FIFO buffer is empty (has no elements), otherwise the function returns false.

NextTxVector()
This function returns a 72-bit vector carrying a single EQ as shown in Figure 143-2. The vector is constructed from the data received from 25GMII over two subsequent 36-bit transfers: the first transfer is on rising TX_CLK25 edge and the second transfer is on the falling TX_CLK25 edge.

PassToGearbox(v)

This function passes a 257-bit vector v to the Gearbox for outputting to the PMA.

Transcode()

This function performs 64B/66B to 256B/257B transcoding and scrambling per 91.5.2.5 and returns the result. It takes an array of four 66-bit blocks a[4] as an argument and returns a 257-bit vector.

142.2.5.4. State Diagrams

142.2.5.4.1. PCS Input process

Note to Editor: per section 142.2.2.1 with changes as noted.

The PCS Input process accepts two consecutive 36-bit transfers from the xMII interface and converts them into a single 72-bit tx_raw vector. The Input process discards all RATE_ADJ_EQs to allow for insertion of FEC parity blocks by the PCS Transmit process (see 142.2.5.4.3). IBI_EQs not required to complete a 256B/257B block at the end of a transmission are also discarded at the Input process. All other 72-bit vectors are encoded into 64B/66B blocks. Four 64B/66B blocks are accumulated, scrambled, and transcoded into a single 256B/257B block and copied to the FEC Encoder. A single bit indicating the accompanying 256B/257B vector has been scrambled is appended to the vector which is then stored in the INPUT_FIFO.

Deleted: functional block

Deleted: block

Deleted: TBD

Deleted: 142.x.x

Deleted: block

Deleted: scrambled

The Nx25G PCS shall perform the Input process as shown in **Error! Reference source not found.**.

Figure 142-6 PCS Input process state diagram

142.2.5.4.2. PCS Framer process

Note to Editor: per section 142.2.2.2 with changes as noted.

The PCS Framer process monitors data from the INPUT_FIFO and transfers it to the TX_FIFO, inserting inter-burst idle (IBI), SP, parity placeholders (PAR_PLACEHLDR), and EBD as appropriate. While the INPUT_FIFO is empty, the PCS Framer process appends IBI to the TX_FIFO. When the INPUT_FIFO first becomes not empty, indicating the beginning of a burst, the SP is appended to the TX_FIFO. Once the complete SP is appended to the TX_FIFO the Framer process begins transferring data from the INPUT_FIFO to the TX_FIFO. When sufficient data for a full FEC payload has been transferred to the TX_FIFO, or the end of the burst is detected as indicated by an empty INPUT_FIFO, the PCS Framer process appends sufficient PAR_PLACEHLDR blocks to the TX_FIFO to allow insertion of the FEC parity codeword into the data stream by the PCS Transmit process. Additional FEC codewords are allowed for until the end of the transmission is indicated by an empty INPUT_FIFO, at which point the Framer process appends the EBD to the TX_FIFO followed by IBI.

The Nx25G PCS shall perform the Framer process as shown in **Error! Reference source not found.**.

Deleted: start of burst delimiter (SBD)

Deleted: end of burst delimiter (

Deleted:)

Deleted: Sync Pattern (

Deleted:)

Deleted: SBD

Deleted: input

Deleted: ,

Deleted: PCS

Figure 142-7 PCS Framer process state diagram

142.2.5.4.3. PCS Transmit process

Note to Editor: per section 142.2.2.3 with changes as noted.

The PCS Transmit process transfers data from the TX_FIFO or FEC Encoder to the PMA. On each transition of the CLK_OUT to True the Transmit process retrieves one 258-bit block of data from the TX_FIFO. If the retrieved 258-bit block is equal to SP[0] and Transmitting is False, indicating the beginning of a transmission, the PMA_SIGNAL.request is set to True indicating that the laser needs to be turned on, and the lower 257-bits of the 258-bit block are sent to the PMA. If the retrieved 258-bit block is EBD and Transmitting is True, indicating the end of a transmission, the PMA_SIGNAL.request is set to False indicating that the laser needs to be turned off, and the lower 257-bits of the 258-bit block are sent to the PMA. If the retrieved 258-bit block is PAR_PLACEHLDR, indicating a FEC parity codeword needs to be inserted in the data stream, 257-bits of the parity are retrieved from the

Deleted: Gearbox

Deleted: ¶

Deleted: Gearbox

Deleted: Gearbox

PARITY_STAGING_BUFFER and sent to the PMA. In all other cases, i.e., normal transmission data, the lower 257-bits of the 258-bit block retrieved from the TX_FIFO are sent to the PMA.

The Nx25G PCS shall perform the Transmit process as shown in **Error! Not a valid bookmark self-reference.**.

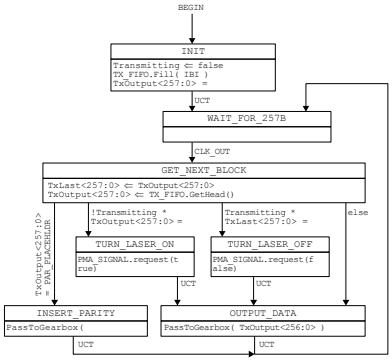

Figure 142–15—PCS Transmit State Diagram

Figure 142-8 PCS Transmit process state diagram

142.3. PCS receive data path

Note to Editor: text from D1.2 142.2.3 with changes

This subclause defines the receive data path of the Nx25G-EPON PCS. In the ONU, the PCS receive data path operates in a continuous mode at a 25.78125 Gb/s rate. In the OLT, the PCS receive data path operates in burst mode at a 25.78125 Gb/s rate (25/25G-EPON, 50/25G-EPON, and 50/50G-EPON or at a 10.3125 Gb/s rate (25/10G-EPON and 50/10G-EPON). The PCS receive data path includes a mandatory LDPC FEC decoder (see 142.3.1). The functional block diagram for the PCS receive data path is shown in Figure 142-2 and consists of the following processes:

— PCS Synchronizer process (see 142.3.5.4.1 and 142.3.5.4.2).
— PCS Receive process (see 142.3.5.4.3).
— PCS BER Monitor process (see 142.3.5.4.4)
— PCS Deframer process (see 142.3.5.4.5)
— PCS Output process (see 142.3.5.4.6)

142.3.1. FEC decoder

Note to Editor: text and figures extracted from 142.2.2.5.

142.3.1.1. LDPC Decoder

Figure 142-9 illustrates the receiver LDPC decoder with shortening/puncturing, de-interleaver data path.

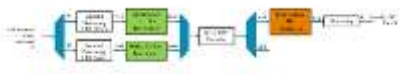

Figure 142–7—FEC decoder

Deleted: FEC encoder

Deleted: Gearbox

Deleted: Gearbox

Deleted: <#>PCS Gearbox process¶
The Gearbox adapts between the 257-bit width of the PCS blocks and the 16-bit width of the PMA interface. It receives the 257-bit blocks from the PCS Transmit process. The Gearbox sends 16 bits of transmit data at a time via the PMA_UNIT-DATA.request primitive. The bits shall be packed into the tx_data vector in sequence with the lowest numbered bit of the block going into the lowest numbered bit of tx_data<15:0> vector (see Figure 142-5 {example only need equivalent to Figure 49–5}).¶

Deleted: direction

Deleted: for

Deleted: {NG-EPON type}

Deleted: function

Deleted: in a continuous mode

Deleted: function

Deleted: may operate

Deleted: , as specified herein ({NG-EPON type, symmetric}),

Deleted: , compliant with Clause {TBD} ({NG-EPON type, asymmetric})

Deleted: For all {NG-EPON types}, the OLT PCS receive function operates in burst mode.

Deleted: function

Deleted: 0.

Deleted: The receive function

Deleted: functional blocks

Deleted: block

Field Code Changed

Deleted: 142.2.3.1

Deleted:

Field Code Changed

Deleted: 142.2.3.2

Deleted: ,

Deleted: <#>FEC Decoder (see 142.2.3.4),¶
256B/257B to 64B/66B Transcoder (see 142.2.3.5), and¶

Deleted: <#>r/Decode block

Deleted: <#>142.2.3.6

Deleted: interleaver/

Figure 142-9 FEC decoder

142.3.1.2. Receive Interleaving

See 142.2.4.4.

142.3.2. 256B/257B to 64B/66B transcoder

Note to Editor: text from 142.2.3.5

The 256B/257B to 64B/66B Transcoder converts one scrambled 256B/257B block received from the PCS Deframer into four consecutive 64B/66B blocks as described in 91.5.3.5 and returns the result to the PCS Output process.

142.3.3. Descrambler

The Nx25G-EPON PCS uses the descrambler specified in 49.2.10.

142.3.4. 64B/66B decoder

Note to Editor: text from 142.2.3.6

See 49.2.11. The 64B/66B decoder shall perform functions specified in the state diagram shown in Figure 49–17. . The Nx25G PCS uses the bit reception order illustrated in Figure 142-10 rather than that shown in Figure 49-6.

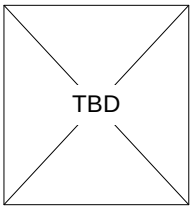

Figure 142-10 PCS receive bit ordering

142.3.5. Receive data path state diagrams

Note to Editor: text in the following sections is combined from 142.2.3.1-142.2.3.1 and 142.2.3.6.1-142.2.3.6.3 and reordered as needed.

142.3.5.1. Constants

EBLOCK_R - see 49.2.13.2.1.

FEC_CW_SZ
TYPE: Integer
The size of the FEC Codeword in bits.
VALUE: {TBD}
Editor's Note (to be removed prior to publication): Note FEC_CW_SZ will likely be defined before this section and could just be cross referenced.

FecFailLimit
TYPE: Integer
The number of FEC decoding failures allowed while in codeword lock before declaring out of lock
VALUE: {TBD}

LBLOCK_R - see 49.2.13.2.1.

MatchTarget
TYPE: Integer
The number of parity delimiters required to transition from a codeword out of lock start to a codeword lock state.
VALUE: {TBD}

PD
TYPE: binary array of {TBD}-bits
The burst delimiter bit pattern found at the beginning of each FEC Parity block.
VALUE: {TBD}

142.3.5.2. Variables

FecDecodeFail
TYPE: Boolean
This clear on read variable indicates the most recent completed FEC codeword decoding failed.

FecDecodeSucceed
TYPE: Boolean
This clear on read variable indicate the most recently completed FEC codeword decoding succeeded.

FecFailCount
TYPE: Integer
This counter track the number of consecutive FEC decoding failures.

Match
TYPE: Boolean
This variable holds the most recent result of the Compare() function.

MatchCount
TYPE: Integer
This counter tracks the number of consecutive successful parity delimiter matched.

rx_buffer
TYPE: binary array
This array hold the sequence of concatenated bits received from the PMA_UNITDATA.indicationprimitive.

rx_coded – see 49.2.13.2.2.

rx_raw – see 49.2.13.2.2.

142.3.5.3. Functions

Compare(v, p)
This function compares bit by bit its two arguments and returns a Boolean ‘true’ if the number of bits that are different is less or equal to the Hamming threshold of {TBD} otherwise the function returns false.

DECODE(rx_coded) - see 49.2.13.2.3.

NextRxValid(prev_rx_raw, next_rx_coded)

This function returns a Boolean indicating whether the *next_rx_coded* vector is valid given the classification of the previously received *prev_rx_raw vector*. The function returns the values according to Table 142–2. Vector classifications used in Table 142–2 are shown in Table 142–3.

NextRxVector()

function which returns the next 66-bit vector from the Descrambler.

Slip(v, bc)

This function removes “bc” bits from the passed array “v”.

142.3.5.4. State Diagrams

142.3.5.4.1. OLT synchronizer process

{text TBD}

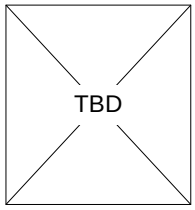

Figure 142- 11 OLT Synchronizer state diagram

142.3.5.4.2. ONU synchronizer process

Note to Editor: text from 142.2.3.2 with changes

Editor’s Note: the text below was adopted from 76.3.3.2 and will need to be aligned with the new synchronizer SD.

The ONU synchronization receives data via the {TBD}-bit PMA_UNITDATA.indication primitive. The synchronizer forms a bit stream from the primitives by concatenating requests with the bits of each primitive in order from rx_data-group<xx> to rx_data-group<xx> (see Figure 142-10). It obtains lock to the FEC codewords within the bit stream using the mechanism shown in Figure 142- 3 and outputs {TBD} codewords to the FEC decoder function.

{TBD description of block handling}

While in codeword lock, the synchronizer copies the FEC-protected bits from each data block and the parity bits of the codeword into an input buffer. When the codeword is complete, the FEC decoder is triggered, and the input buffer is freed for the next codeword.

When in codeword lock, the state diagram continues to check for sync header validity. If 16 or more sync headers in a codeword pair (62 blocks) are invalid, then the state diagram deasserts codeword lock. In addition, if the *persist_dec_fail* signal becomes set, then codeword lock is deasserted (this check ensures that certain false-lock cases are not persistent.)

The ONU Synchronizer shall implement the state diagram as depicted in Figure 142-12.

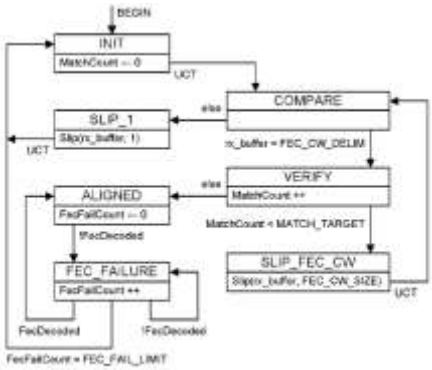

Figure 142–16—ONU Synchronizer state diagram

Figure 142-12 ONU Synchronizer state diagram

142.3.5.4.3. PCS Receive process

142.3.5.4.4. PCS BER monitor process(?)

142.3.5.4.5. PCS Deframer process

142.3.5.4.6. PCS Output process

The OLT and the ONU shall implement the Receive/Decode process as depicted in Figure 142-13.

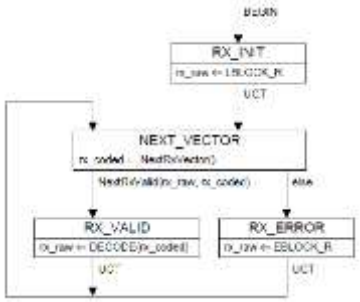

Figure 142-13 PCS Output state diagram

Note to Editor; above fig was 142-17 Transmit/Encode SD

142.4. Nx25G-EPON PMA

Note to Editor: copy D1.2 142.3 and all remaining sections without changes other than renumbering.