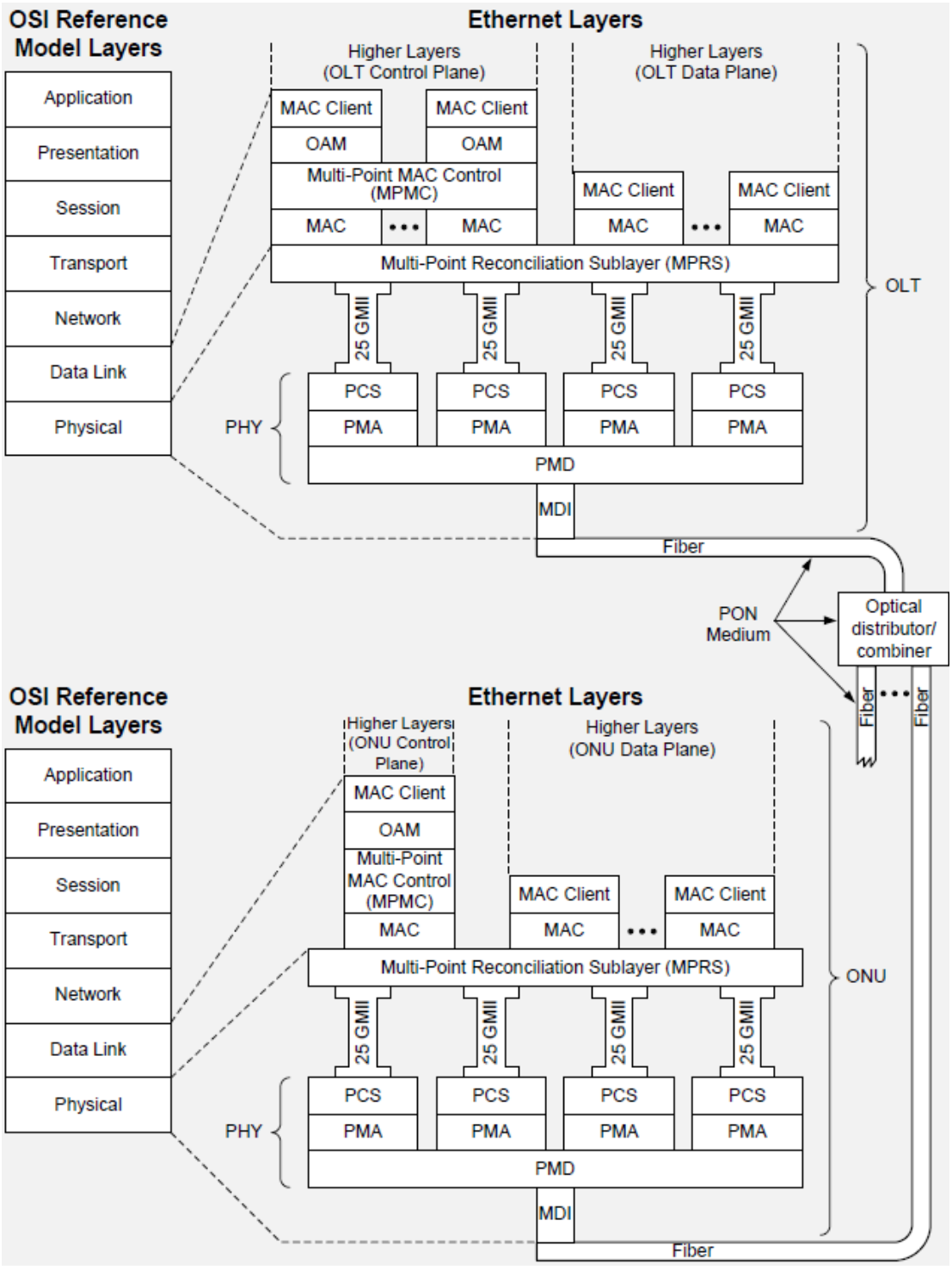
143. Multi-Point Reconciliation Sublayer (MPRS) for 100G-EPON

143.1 Overview

This clause describes the Multi-Point Reconciliation Sublayer (MPRS) used with {100Gtype} 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.

This subclause extends Clause 46 and Clause 105 to enable multiple MACs to interface with multiple Physical Layers, and to enable data links with one data rate (e.g., 10 Gb/s) in one direction but another (e.g., 25 Gb/s) in the opposite direction. The number of MACs supported is limited only by the implementation. It is acceptable for only one MAC to be connected to this Reconciliation Sublayer. Figure 143–1 and Figure 143–2 show the relationship between this RS and the ISO/IEC OSI reference model. The mapping of 25GMII/XGMII signals to PLS service primitives is described in 106.1.7.5 for 25GMII AND 46.1.7 for XGMII with exceptions noted herein.



kramer_3ca_2_0117.pdf slide 14 or similar (HB Motion #6)

Figure 143–1—100G-EPON Layering diagram

kramer_3ca_2_0117.pdf slide 14 or similar

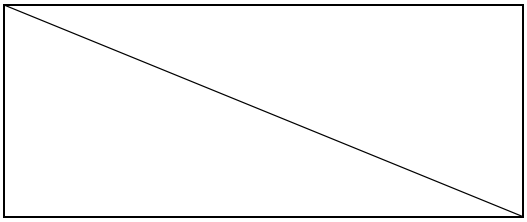


Figure 143–2—Figure placeholder

143.1.1 Summary of major concepts

This subclause applies to the interface between the MAC and PHY in an OLT or an ONU. The physical implementation of the interface is primarily intended to be chip-to-chip, but may also be used as a logical interface between ASIC logic modules within an integrated circuit. These interfaces are used to provide media independence, so that an identical media access controller may be used with all 100G-EPON PHY types.

143.1.1.1 25Gb/s, 50Gb/s, and 100Gb/s operation

{text}

143.1.1.2 MAC to MII binding

A successful registration process, described in 144.x.x, results in the assignment of a value to the LLID variable associated with a MAC. This may be one of many MACs in an OLT or ONU. The LLID variable is used to identify a packet transmitted from that MAC and how received packets are directed to that MAC. The RS in the OLT shall operate in unidirectional mode as defined in 66.4.

NOTE: in the above para the MODE variable was eliminated and the concept of a single LLID per ONU removed.

MAC binding to the XGMII in a 25/10G-EPON OLT or ONU is as described in 77.1.2.

Prior generation EPON reconciliation sublayers had a many-to-one relationship between MAC and MII interface. For 25G-EPON this is still the case between MACs and the single 25GMII. However, for other 100G-EPON systems there may be a many-to-many relationship between a MAC and the two or four 25GMII interfaces. Similarly, in prior generations, only one PLS_DATA.request was active which is still true for 25G-EPON systems. For other 100G-EPON systems there may be one, two or four PLS_DATA.request interfaces active at any one time. See 76.2.3 for additional information on 10G-EPON.

In the transmit direction of all 100G-EPON systems except the 25/10G-EPON ONU, the RS maps an active PLS_DATA.request to the 25GMII signals XGMII signals (TXD[x]<31:0>, TXC[x]<3:0>, and TX_CLK) as indicated by the most recent MPRS Control Primitive (see 143.1.1.3). The RS inserts the values of the transmitting MAC's LLID variable in the envelope header as described in 143.1.1.3.

In the 25/10G-EPON ONU transmit direction, the RS maps the active PLS_DATA.request to the XGMII signals (TXD10<31:0>, TXC10<3:0>, and TX_CLK10) and the RS replaces octets of preamble with the values of the transmitting MAC's MODE and LLID variables (see 76.2.6.1).

143.1.1.3 Transmission Envelope

Transmitted data in 100G-EPON is contained in Transmission Envelopes. A Transmission Envelope encapsulates data from one LLID, either a user LLID (ULID) or a PON LLID (PLID). A Transmission Envelope can contain, at most, two partial frames (one at the beginning and one end of the envelope) and any number of whole frames. Each Transmission Envelope begins with an eight byte header (see Figure 143–3.

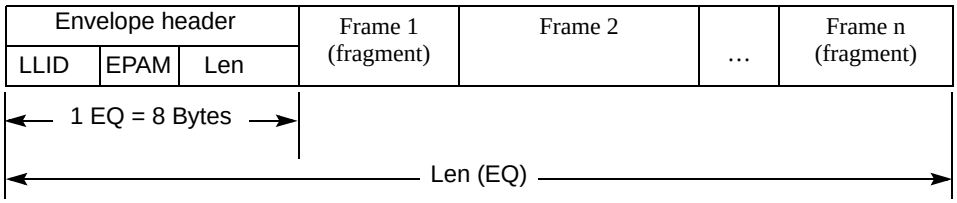


Figure 143–3—Transmission Envelope structure

143.1.1.3.1 Transmission Envelope header

The Transmission Envelope header, as illustrated in Figure 143–4, includes two successive 36-bit transfered over the 25GMII interface. Each 36-bit transfered includes four control bits followed by 32 information bits. These information bits contain the Envelope Header ordered sets (/EHO1/ or /EHO2/), an LLID field, an Envelope Position Alignment Marker (EPAM) field or an Envelope length field. The LLID field is set to the value of the LLID variable of the MAC associated with the data in the envelope. The EPAM is used by the receiving MPRS to remove any timing skew that may have occurred during the transmission of the envelope from the transmitting MPRS to the receiving MPRS. The envelope length represents the number of Envelope Quanta (EQ) in the envelope, including the header. An EQ contains eight bytes of information so the length of the Envelope Header is one EQ.

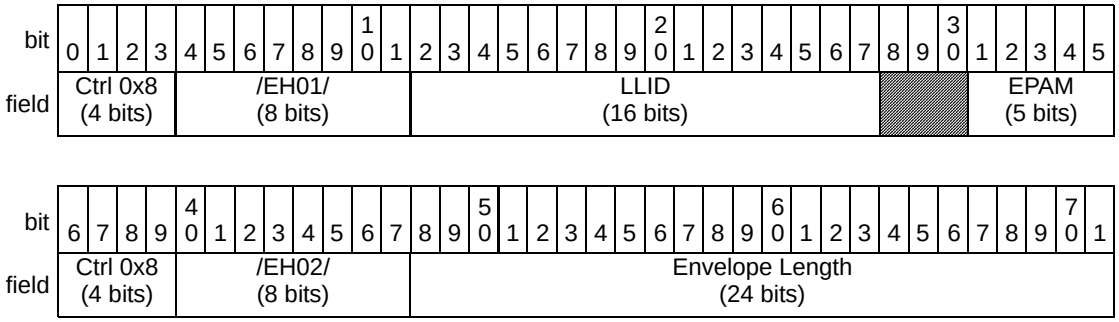


Figure 143–4—Transmission Envelope header format

143.1.1.3.2 Overlapping Envelopes

A single LLID may be have an active Transmission Envelope in more than one channel (i.e., 25GMII). Data is interleaved in multiple envelopes allowing for up to 100 Gb/s transmission by a single MAC (LLID). The interleaving unit is EQ (see Figure 143–5).

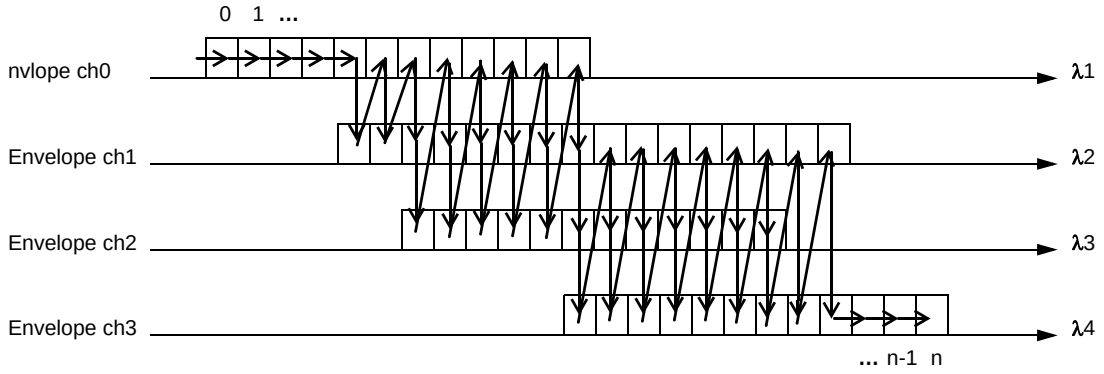


Figure 143–5—Overlapping Transmission Envelopes

143.1.2 Channel bonding in 100G-EPON

{text}

143.2 100G-EPON Requirements

143.2.1 Delay constraints

The Multi-Point Control Protocol (MPCP) relies on strict timing based on the distribution of timestamps. The actual delay is implementation dependent but an implementation shall maintain a combined delay variation through RS of no more than 1 envelope_quantum (see 144.x.x.x) so as not to interfere with the MPCP timing.

NOTE: in the above 1 TQ (changed to 1 EQ) was from Cl 76 and applied to combined RS, PCS, & PMA. A revised figure may be needed.

143.2.2 TBD

NOTE: IEEE style guide precludes single subclause levels “12.1 ... Clauses and subclauses should be divided into further subclauses only when there is to be more than one subclause.”. If no additional subclauses to 143.2 are required 143.2.1 should be folded into 143.1.

143.3 MPRS Control Primitives

The 100G-EPON RS inputs the MPRS_CTRL[ch].request primatives from the MPCP and outputs to the MPCP the MPRS_CTRL[ch].indication primitives (see Figure 143–6 and Figure 143–7).

143.3.1 MPRS_CTRL[ch].request(LinkID, epam, env_length) primitive

The MPCP requests the MPRS to transmit the next envelope using the MPRS_CTRL[c].request(LinkID, epam, env_length) primitive. This opens an envelope on a channel ch for the LLID specified by LinkId with a length (in EQs) of env_length. If all channels are idle the MPRS EnvPam variable is set to the value of epam (see EnvHeader function definition in section 143.5.7).

143.3.2 MPRS_CTRL[ch].indication(cw_left) primitive

The Input process requests the next envelope from the MPCP after the completion of the previous envelope using the MPRS_CTRL[ch].indication(cw_left) primitive. This primitive indicates to the MPCP that the MPRS is available for the next envelope and shows the available space in the current FEC codeword. In absence of an active envelope, the MPRS_CTRL[ch].indication(cw_left) primitive is generated continuously on every IN_CLK transition. The MPCP can decide whether to issue a new envelope immediately adjacent to the previous envelope (for envelopes that are expected to be packed in the same grant), or wait for the start of next FEC codeword (for envelopes that are expected to be in a separate grant). After the gap from the last envelope exceeds GRANT_MARGIN (the time required to turn on the laser), every MPRS_CTRL[ch].indication(cw_left) will indicate that a full FEC codeword is available (cw_left = FEC_CODEWORD_SIZE), implying that the next envelope will start with a new FEC codeword.

NOTE: the paras above were largely taken from kramer_3ca_2c_0916 slide 27. We might want to move this to he MPCP clause and just ref MPCP

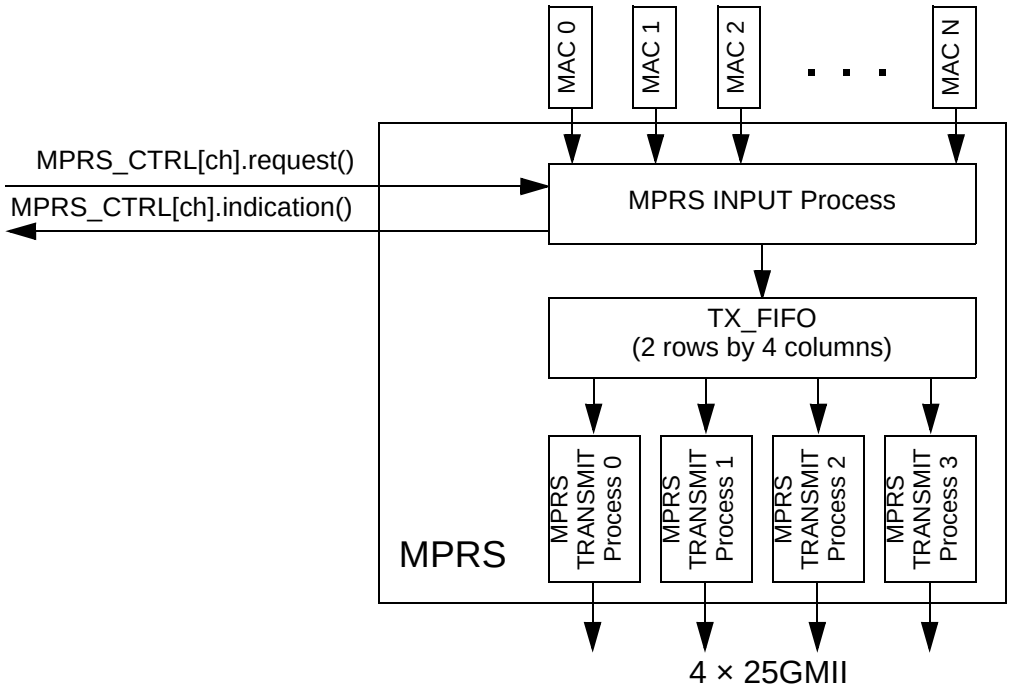


Figure 143–6—100G-EPON MPRS transmit functional block diagram

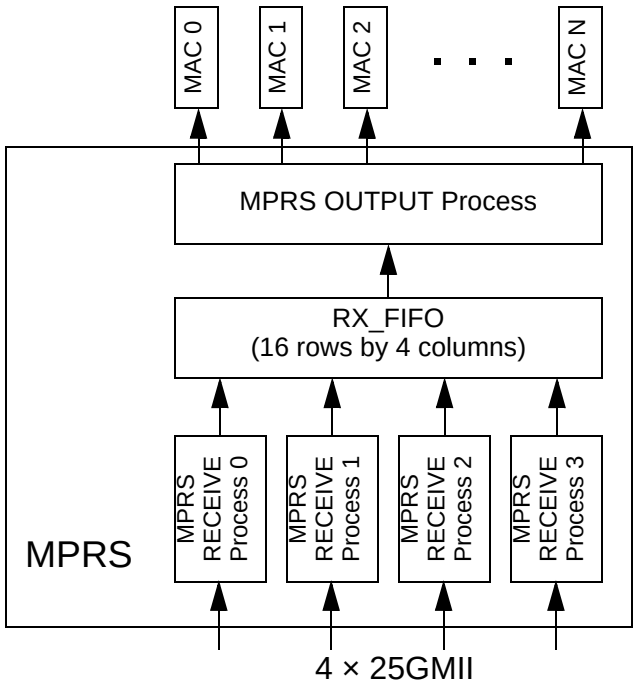


Figure 143–7—100G-EPON MPRS receive functional block diagram

143.4 Multi-speed Media Independent Interface

In symmetric architectures, the same interface is used to transfer data between the RS and the PCS. For the 100G-EPON architectures, the 25GMII is the interface used to transfer data between the MPRS to the PCS. When using a 25/10G-EPON architecture, a combination of both 25GMII and XGMII is needed in order to support transmission and reception at different speeds. Through the parallel use of the GMII and XGMII or multiple 25GMIIs, the following modes are supported:

— Asymmetric-rate operation (25 Gbps downstream and 10 Gbps upstream) for transmit and receive data paths at the OLT, utilizing transmit path functionality of the 25GMII defined in Clause 106 and receive path functionality of the XGMII defined in Clause 46.

— Asymmetric-rate operation (25 Gbps downstream and 10 Gbps upstream) for transmit and receive data paths at the ONU, utilizing transmit path functionality of the XGMII defined in Clause 46 and receive path functionality of the 25GMII defined in Clause 106.

— Asymmetric-rate operation (50 Gbps or 100 Gbps downstream and 25 Gbps or 50 Gbps upstream, respectively) for transmit and receive data paths at the OLT, utilizing transmit path functionality of the two or four and receive path functionality of one or two 25GMIIs defined in Clause 106.

— Asymmetric-rate operation (50 Gbps or 100 Gbps downstream and 25 Gbps or 50 Gbps upstream, respectively) for transmit and receive data paths at the ONU, utilizing transmit path functionality of one or two and receive path functionality of two or four 25GMIIs, , defined in Clause 106.

— Symmetric-rate 25 Gbps operation for transmit and receive data paths, utilizing the functionality of a single 25GMII defined in Clause 106.

— Symmetric-rate 50 Gbps operation for transmit and receive data paths, utilizing the functionality of two 25GMII defined in Clause 106.

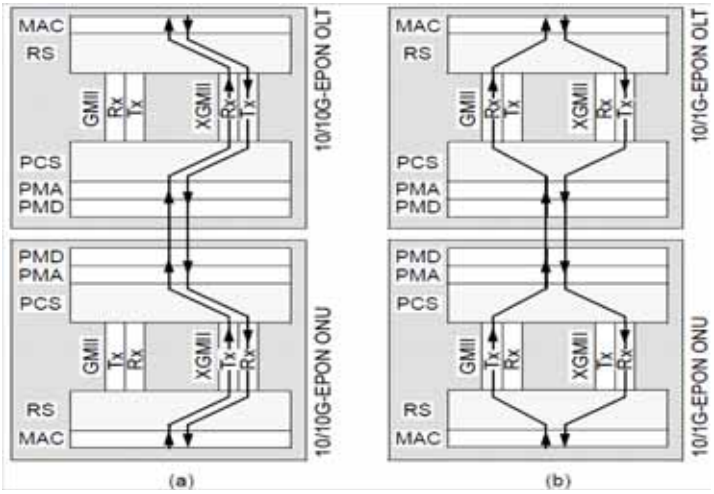
— Symmetric-rate 100 Gbps operation for transmit and receive data paths, utilizing the functionality of four 25GMII defined in Clause 106.

— Coexistence of various ONU types by utilizing different data paths within the OLT.

143.4.1 25/10G-EPON

At the OLT, the transmit path uses 25GMII signals TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25 of the transmit channel 0, while the receive path uses RXD10<31:0>, RXC10<3:0> and RX_CLK10 of the XGMII. At the ONU, the transmit path uses XGMII signals TXD10<31:0>, TXC10<3:0> and TX_CLK10, while the receive path uses 25GMII signals RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0].

Figure 143–8 depicts the data paths used in 25/10G-EPON.



For Reference only (from Fig 76-3). Do we want a similar figure for 100G or modify the figure adopted from kramer_3ca_2_0117.pdf slide 15?

Figure 143–8—Figure placeholder

143.4.2 100G-EPON

Figure 143–9 depicts the data paths used in 100G-EPON.

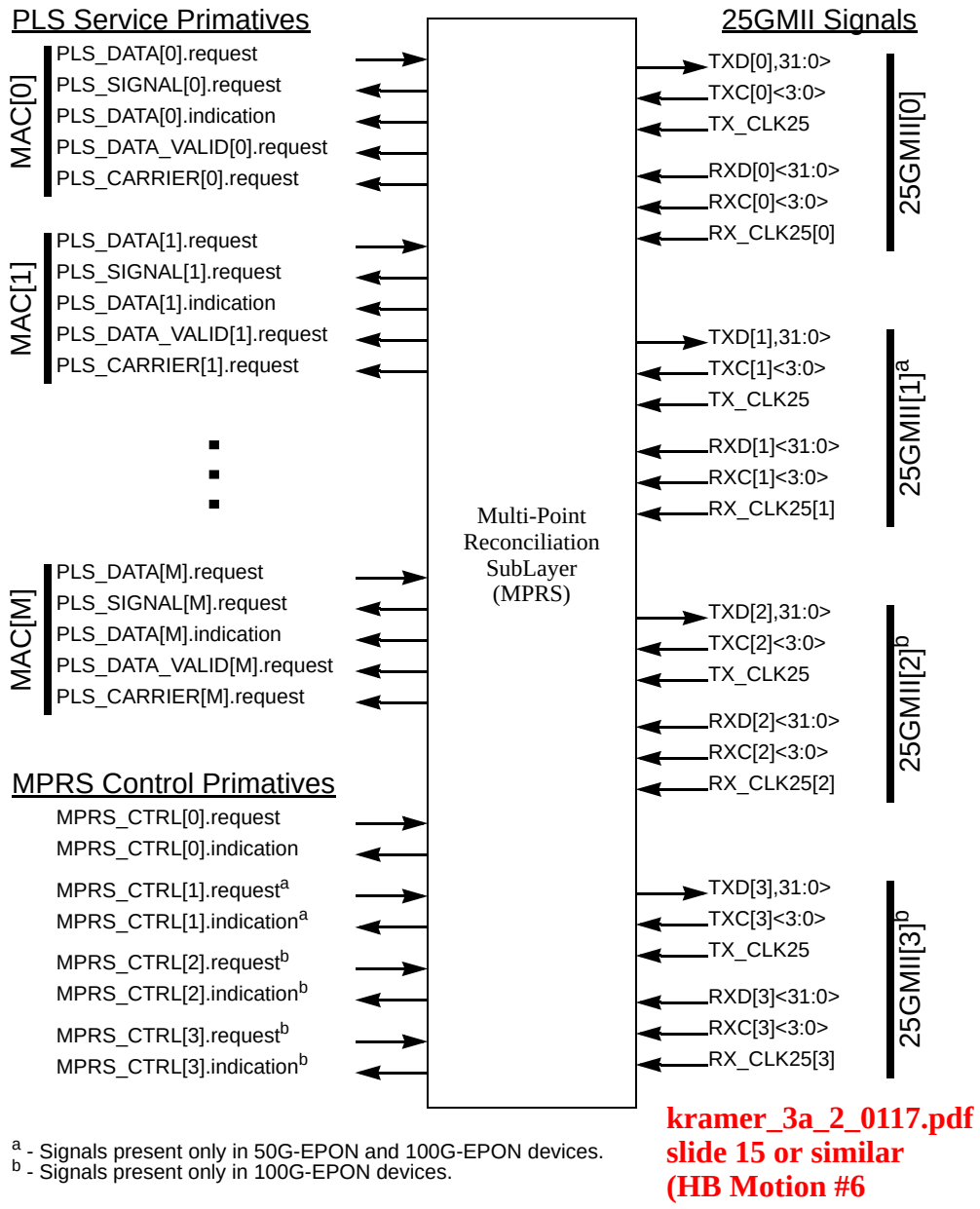


Figure 143–9—100G-EPON MPRS inputs and outputs

143.4.3 Multi-rate mode

To support coexistence of 25G-EPON and 10G-EPON ONUs on the same outside plant, the OLT may optionally support dual-rate mode. Dual-rate mode supports transmission and reception at both 25 Gb/s and 10 Gb/s. When operating in a dual-rate mode, a combination of 25GMII and XGMII data paths are used for transmission and reception. Figure 143–8 depicts the data paths used in an OLT operating in a dual-rate mode.

To support operation at sub-rate 25G-EPON, 50G-EPON and asymmetric combinations of 25G, 50G, and 100G-EPON an OLT or ONU may support one, two, or four 25GMII data paths.

143.4.4 25GMII structure

The 25GMII structure of each of the four 25GMII interfaces in a 100G-EPON system is identical to the XGMII structure specified in 46.1.6.

NOTE: this wording is from Cl 106, we should rationalize wording for 25GMII & XGMII structure.

143.4.5 XGMII structure

The XGMII structure is discussed in 46.1.6, and Figure 46–2 depicts a schematic view of the RS inputs and outputs.

NOTE: this wording is from Cl 76, we should rationalize wording for 25GMII & XGMII structure.

143.4.6 Mapping of 25GMII and XGMII primitives

The mapping of 25GMII/XGMII signals to the PLS_DATA.request and PLS_DATA.indication primitives is described in 46.1.7.1. Additional details are provided in Table 143–1, which shows the mapping of PLS_DATA.request primitives to transmit interface signals for different types of OLTs and ONUs. Table 143–2 shows the mapping of PLS_DATA.indication primitives to receive interface signals for different types of OLTs and ONUs.

NOTE: how exhaustive in these two tables do we want to get??? 25/50G? 50/100G?

Table 143–1—Mapping of PLS_DATA.request primatives

MAC location	MAC operating speed	Transmit interface	signals
OLT	10/25G-EPON	25GMII[0]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25
OLT	25G-EPON	25GMII[0]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25
OLT	50G-EPON	25GMII[0] 25GMII[1]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25 TXD[1]<31:0>, TXC[1]<3:0>
ONU	100G-EPON	25GMII[0] 25GMII[1] 25GMII[2] 25GMII[3]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25 TXD[1]<31:0>, TXC[1]<3:0> TXD[2]<31:0>, TXC[2]<3:0> TXD[3]<31:0>, TXC[3]<3:0>
ONU	10/25G-EPON	XGMII	TXD10<31:0>, TXC10<3:0> and TX_CLK10
ONU	25G-EPON	25GMII[0]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25
ONU	50G-EPON	25GMII[0] 25GMII[1]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25 TXD[1]<31:0>, TXC[1]<3:0>
ONU	100G-EPON	25GMII[0] 25GMII[1] 25GMII[2] 25GMII[3]	TXD[0]<31:0>, TXC[0]<3:0> and TX_CLK25 TXD[1]<31:0>, TXC[1]<3:0> TXD[2]<31:0>, TXC[2]<3:0> TXD[3]<31:0>, TXC[3]<3:0>

Table 143–2—Mapping of PLS_DATA.indication primatives

MAC location	MAC operating speed	Receive interface	signals
OLT	10/25G-EPON	25GMII[0]	RXD10<31:0>, RXC10<3:0> and RX_CLK10
OLT	25G-EPON	25GMII[0]	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0]
OLT	50G-EPON	25GMII[0] 25GMII[1]	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0] RXD[1]<31:0>, RXC[1]<3:0> and RX_CLK25[1]
ONU	100G-EPON	25GMII[0] 25GMII[1] 25GMII[2] 25GMII[3]	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0] RXD[1]<31:0>, RXC[1]<3:0> and RX_CLK25[1] RXD[2]<31:0>, RXC[2]<3:0> and RX_CLK25[2] RXD[3]<31:0>, RXC[3]<3:0> and RX_CLK25[3]
ONU	10/25G-EPON	XGMII	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0]
ONU	25G-EPON	25GMII[0]	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0]
ONU	50G-EPON	25GMII[0] 25GMII[1]	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0] RXD[1]<31:0>, RXC[1]<3:0> and RX_CLK25[1]
ONU	100G-EPON	25GMII[0] 25GMII[1] 25GMII[2] 25GMII[3]	RXD[0]<31:0>, RXC[0]<3:0> and RX_CLK25[0] RXD[1]<31:0>, RXC[1]<3:0> and RX_CLK25[1] RXD[2]<31:0>, RXC[2]<3:0> and RX_CLK25[2] RXD[3]<31:0>, RXC[3]<3:0> and RX_CLK25[3]

143.5 Transmit functional specification for multiple MAC and MII EPON

143.5.1 Conventions

The notation used in the state diagrams in this subclause 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 counter indicates it is to be incremented by 1. The notation – – after a counter indicates it is to be decremented by 1. The notation – = after a counter indicates that the counter value is to be decremented by the following value. The notation + = after a counter 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. Variables are unsigned unless stated otherwise.

143.5.2 Constants

FEC_CODEWORD_SIZE
TYPE: integer
Value: {TBD}
The size of the FEC codeword in EQs.

FEC_PARITY_SIZE
TYPE: integer
Value: {TBD}
The size of the FEC parity word in EQs.

NO_ENV_CODE

TYPE: {TBD}
Value: {TBD}
A control code written to the TX_FIFO when no active envelope exists (i.e., the channel was not assigned to any LLID).

PARITY_PLACEHLDR

TYPE {TBD}
Value: {TBD}
A control code written to the TX_FIFO when data from MAC should be paused to allow FEC parity insertion. In the PCS sublayer, PARITY_PLACEHLDR code is overwritten by the calculated parity values.

143.5.3 Counters

{text}

143.5.4 Timers

{text}

143.5.5 Variables

ch

TYPE: 2-bit integer
The ch variable represents the 25GMII channel and TX_FIFO buffer column currently being read by the Transmit Process.

CwdLeft[c]

TYPE: {TBD}-bit integer
The CwdLeft variables track the remaining space in the FEC codeword for channel c. Upon filling the payload portion of the codeword (CwdLeft = FEC_PARITY_SIZE), the input process will defer taking more data from the MAC to allow FEC parity to be inserted.

EnvLeft[wCol]

TYPE: 25-bit signed integer
If positive EnvLeft indicates the length remaining in the current envelope for channel wCol, if negative this variable represents the number of EQ periods since the end of the last envelope on the channel.

EnvPam

TYPE: 5-bit integer
The EnvPam variable represents row index in the RX_FIFO into which the received data is to be written, its primary function is to remove skew accumulated during transport between two or more channels from a single transmitter. This variable is set when all channels are idle and is loaded into the Envelope Header using the EnvHeader function (see 143.5.7). If one or more channels are active the variable is incremented after all TX_FIFO columns have been read (i.e., each IN_CLK).

GRANT_MARGIN

TYPE: {TBD}-bit integer
The GRANT_MARGIN variable represents the amount of time, in EQs, necessary to turn on the laser.

IN_CLK

TYPE: Boolean

The IN_CLK clear on read variable is set to True on each positive edge of the TX_CLK25 signal.

LinkID[wCol]

TYPE: 16-bit integer

The LinkID[wCol] variables represent the MAC (LLID) being transferred to the TX_FIFO column wCol by the Input Process.

rRow

TYPE: 1-bit integer

The variable rRow represents the row in the TX_FIFO buffer currently being read by the Transmit Process. The value of this variable is synchronized to wRow and is equal wRow - 1.

TX_CLK[ch]

TYPE: Boolean

Each TX_CLK[ch] clear on read variable is set to True on each edge, positive and negative, of the TX_CLK25 signal.

TX_FIFO[col][row]

TYPE: 72-bit binary array

The TX-FIFO buffer is used to transfer information between the Input Process and the Transmit Process. Each cell in this buffer stores one EQ (a 72-bit vector) of information. The buffer has N columns (col) and two rows (row). The number of columns is dependent on the number of channels supported. For 100 Gb/s devices N = 4, for 50 Gb/s devices N = 2 and for 25 Gb/s devices N = 1. The buffer is filled in a cyclic pattern row-by-row. The source LLID for each cell is determined by the MPRS_CTRL[].request().

wCol

TYPE: 2-bit integer

The wCol variable represents the column in TX_FIFO buffer currently being written by the Input Process. Each column corresponds to a separate transmission channel, i.e., a separate 25GMII interface.

wRow

TYPE: 1-bit integer

The variable wRow represents the row in the TX_FIFO buffer currently being written by the Input Process. The value of rRow is synchronized to this variable and is equal to wRow - 1.

143.5.6 Messages

{text}

143.5.7 Functions

EnvHeader(wCol, epam)

The EnvHeader() function returns an EQ representing an envelope header, which has the format show in Table 143–3.

```
EnvHeader(int2 col, int5 epam) //col – 2 bits; epam – 5 bits
{
   EQ hdr;
   if( EnvLeft[col+1] == GRANT_MARGIN &&
      EnvLeft[col+2] == GRANT_MARGIN &&
      EnvLeft[col+3] == GRANT_MARGIN ) EnvPam = epam;
   hdr<23:8> = LinkId[col];              //LLID
   hdr<59:36> = EnvLeft[col];            //EnvLength
   hdr<4:0> = EnvPam;                    //EPAM
   return hdr;
```

Table 143–3—Envelope Header EQ

Bits	Value	Description
0-3	0x0	Control bits corresponding to TXC<3:0>
4-11	TBD	Envelope Ordered Set 1
12-27	varies	LLID value for the given envelope
28-30	0x000	Reserved (pad)
31-35	varies	Envelope Position Alignment Marker (Number of bits matches the size of wRow)
36-39	0x0	Control bits corresponding to TXC<3:0>
40-47	TBD	Envelope Ordered Set 1
48-71	varies	Length of envelope (in EQ)

}

GetMacBlock(LinkID[wCol])

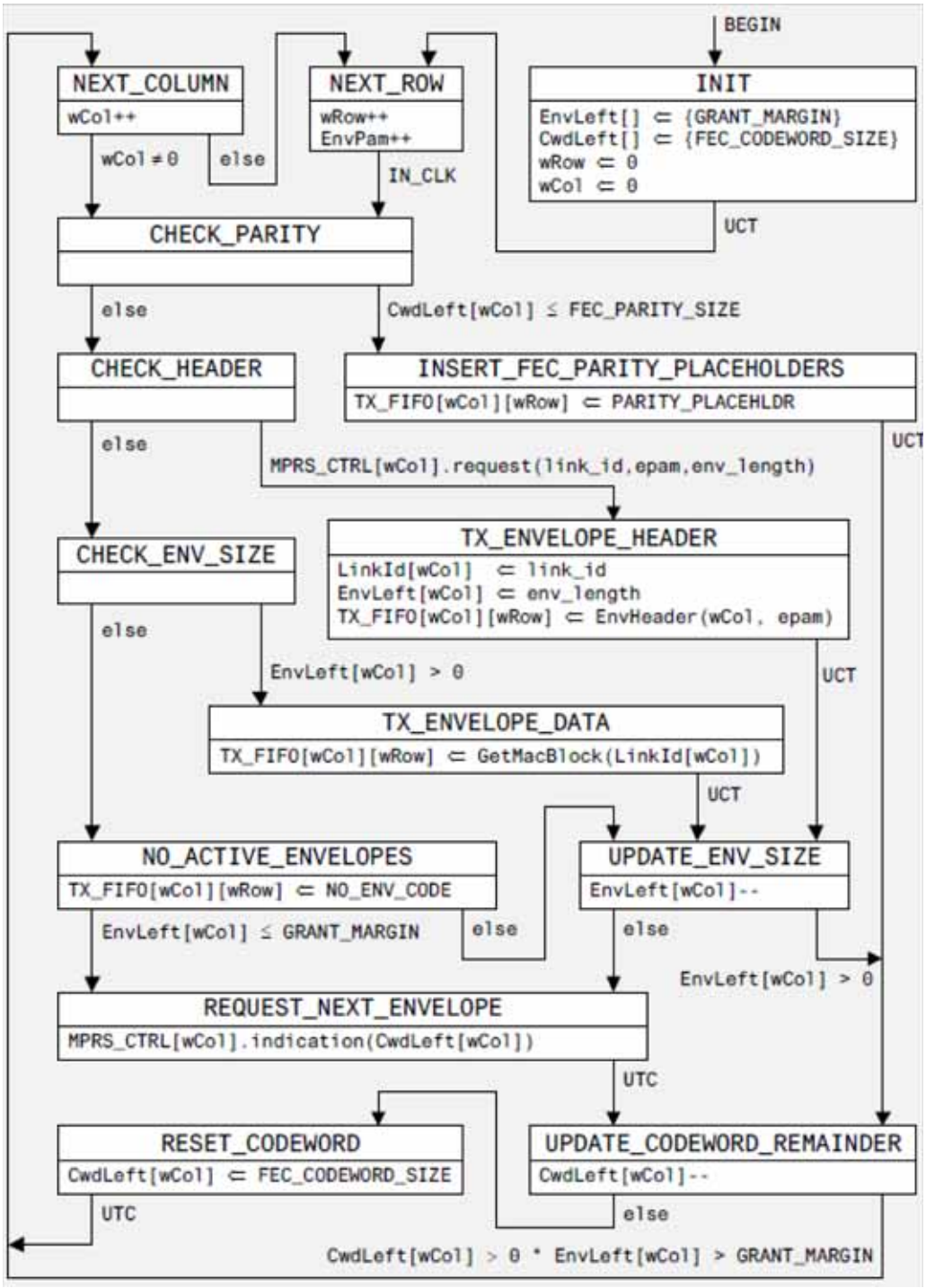
The GetMacBlock function retrieves eight octets (64 bits) of data from a MAC identified by the LinkID parameter. The function returns an EQ that contains both the data and the corresponding eight control bits. If no data is available from the MAC for a particular byte the function returns IDLE control code for that byte. This is a blocking function that returns control to the calling routine after 64 successive invocations of PLS_DATA.request() primitive.

143.5.8 State Diagrams

143.5.8.1 Input process

The 100G-EPON MPRS Input Process shall implement the state diagram as depicted in Figure 143–1.

The Input Process accepts data from a MAC interface and transfers that data to the TX_FIFO one EQ at a time. The process prepends a header to each envelope. Only one instance of the process is needed. The Input Process fills one full row (four columns) in the TX_FIFO buffer on each cycle of IN_CLK (IN_CLK is half the effective rate of TX_CLK). In case of overlapping envelopes, blocks in multiple columns will be retrieved from the same MAC. The process keeps track of the envelope sizes for each LLID and won't exceed the allowed number of EQs for a given envelope. The process adjusts the MAC rate to account for FEC parity insertion in the PCS.



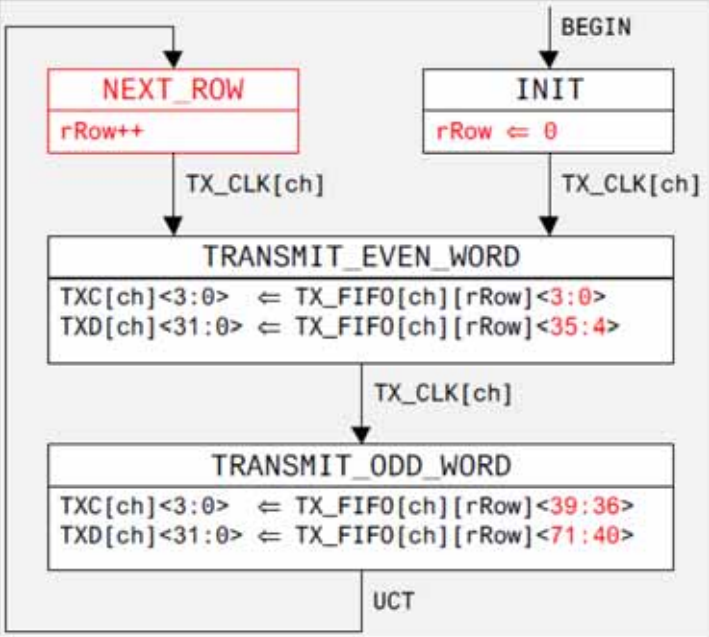
kramer_3ca_1a_1116.pdf slide 4 or similar (SA Motion #4)

Figure 143–10—MPRS Input Process state diagram

143.5.8.2 Transmit process

The 100G-EPON Transmit Process shall implement the state diagram as depicted in Figure 143–1. One instance of the state diagram is instantiated for each 25GMII.

The Transmit Process outputs one 36-bit vector (TXD[x]<31:0> + TXC[x]<3:0> to it's associated 25GMII interface on each edge of the TX_CLK25 signal. There is one instantiation of the Transmit Process for each channel implemented in the device. The main function of the process is to transmit a column of one row from TX_FIFO buffer on an existing channel. The Transmit Process is synchronized to TX_CLK25.



kramer_3ca_1a_1116.pdf slide 13 or similar (SA Motion #4)

Figure 143–11—MPRS Transmit Process state diagram

143.6 Receive functional specification for multiple MAC and MII EPON

143.6.1 Conventions

See 143.5.1.

143.6.2 Constants

NO_ENV_CODE
See 143.5.2.

PARITY_PLACEHLDR
See 143.5.2.

143.6.3 Counters

{text}

143.6.4 Timers

{text}

143.6.5 Variables

ch

TYPE: 2-bit integer

The ch variable represents the 25GMII channel and RX_FIFO buffer column currently being written by the Receive Process.

EnvLeft[rCol]

TYPE: 24-bit integer

The EnvLeft variable represents the number of EQs that remain to be transferred to a MAC in the current envelope on channel rCol.

LinkID[rCol]

TYPE: 16-bit integer

The LinkID[rCol] variables represent the RX_FIFO column rCol being transferred to the MAC (LLID) via the Output Process.

OUT_CLK

TYPE: Boolean

The OUT_CLK clear on read variable corresponds is set to Ture on each positive edge of TX_CLK and runs at half the frequency of TX_CLK.

OutEQ

TYPE: 72-bit binary

The OutEQ variable represents the most recent EQ removed from a single cell of the RX_FIFO buffer.

rCol

TYPE: 2-bit integer

The rCol variable represents the column in RX_FIFO buffer currently being read by the Output Process. Each column corresponds to a separate reception channel, i.e., a separate 25GMII interface.

rRow

TYPE: 5-bit integer

The rRow variable represents the row index in the RX_FIFO buffer currently being read by the Output Process..

RX_CLK[ch]

TYPE: Boolean

The RX_CLK[ch] clear on read variables are set to True on each edge of RX_CLK25[ch] signal and represent the continuous clock that provides the timing reference for the transfer of the RXC[ch]<3:0> and RXD[ch]<31:0> signals received on the 25GMII channel ch.

RX_FIFO[col][row]

TYPE: 72-bit binary array

The RX-FIFO buffer is used to transfer information between the Receive Process and the Output Process. Each cell in this buffer stores one EQ (a 72-bit vector) of information. The buffer has N columns (col) and M rows (row). The number of columns is dependent on the number of channels supported. For 100 Gb/s devices N = 4, for 50 Gb/s devices N = 2 and for 25 Gb/s devices N = 1. The size of M is application specific. The buffer is filled in a cyclic pattern row-by-row.

RxEQ

TYPE: 72-bit binary

The RxEQ variable represents the most recent EQ received from a 25GMII interface.

TX_FIFO[c][r]

TYPE: 72-bit binary

The TX-FIFO variables form the four column (c) by two row (r) transmit buffer between the Input Process and the Transmit Process.

wCol

TYPE: 2-bit integer

wRow

TYPE: 5-bit integer

The variable wRow represents the row in TX_FIFO buffer currently being written by the Receive Process.

143.6.6 Messages

{text}

143.6.7 Functions

OutputToMac(LinkID[rCol], OutEQ)

The OutputToMac(LinkID[rCol], OutEQ) function transfers the eight information bytes in the OutEQ parameter to the MAC associated with the LLID value of LinkID[rCol] per the eight control bits in the OutEQ parameter.

IsMisaligned(RxEQ)

The IsMaisaligned(RxEQ) function returns true is the parameter RxEQ is misaligned, i.e., shifted by half-EQ.

```
bool IsMisaligned( eq )
{
   return(( eq<39:36> == 0xF AND     // Misaligned NO_ENV_CODE
      eq<71:40> == NO_ENV_CODE_LO )  // … s.b. NO_ENV_CODE_HI
      OR
      ( eq<39:36> == 0x8 AND         // Misaligned Env. Header
      eq<47:40> == OCODE1 ));        // … s.b. OCODE2
}
```

IsHeader(RxEQ)

The IsHEader() function returns true if the parameter RxEQ represents an envelope header. A header is an ordered set (see 802.3, clause x.x.x)

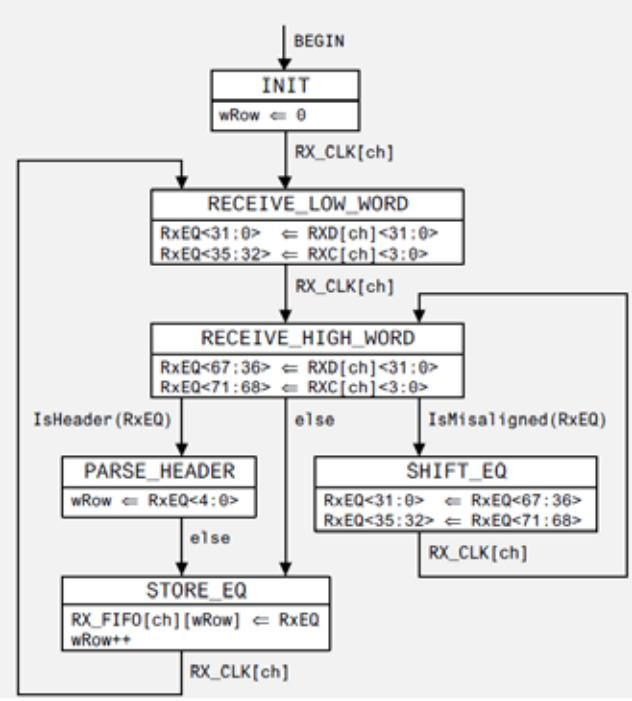
```
bool IsHeader(RxEQ)
{
   return( RxEQ<3:0> == 0x8 AND     // Control bits (low half)
      RxEQ<39:36> == 0x8 AND        // Control bits (high half)
      RxEQ<11:4> == OCODE1 AND      // Env. Header O-code #1 (TBD)
      RxEQ<47:40> == OCODE2 );      // Env. Header O-code #2 (TBD)
}
```

143.6.8 State Diagrams

143.6.8.1 Receive process

The 100G-EPON Receive Process shall implement the state diagram as depicted in Figure 143–1.

This process forms an EQ from two 25GMII transfers. The process first verifies proper alignment of the EQ and, if misaligned, shifts the input by half-EQ. No other error checking is performed by this process. When an Envelope Header is received, the EPAM field is extracted and used as a write position for RX_FIFO buffer. Because the phase of the receive clock (RX_CLK[ch]) in every channel is different due to different ONUs and transport skew a separate instance of the Receive Process is required for each channel implemented.



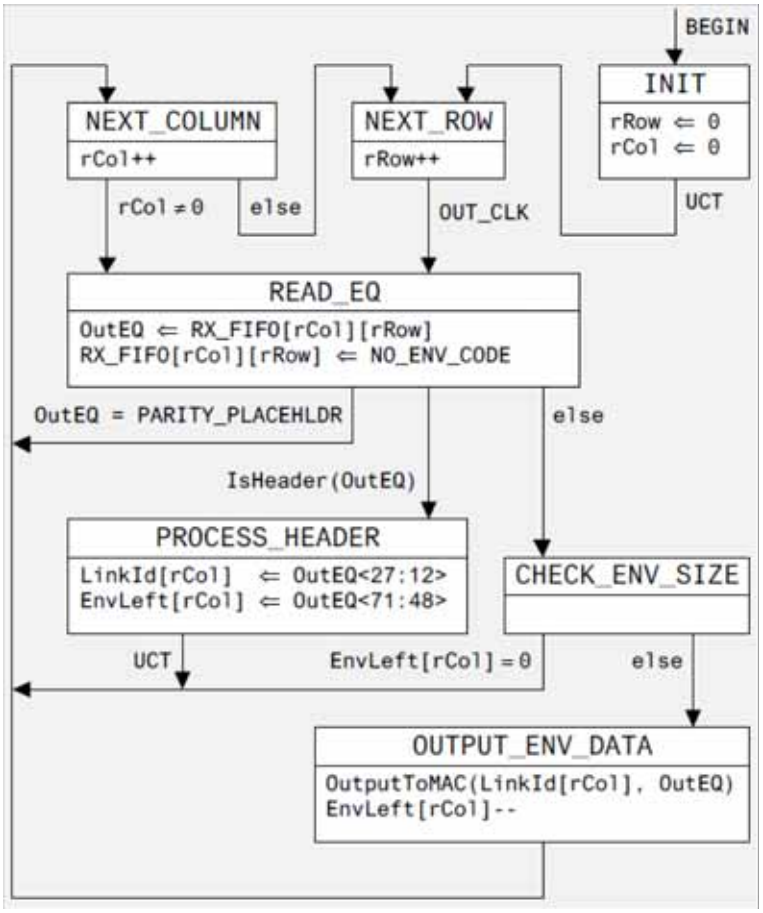
kramer_3ca_3b_1116.pdf slide 3 or similar (SA Motion #6)

Figure 143–12—MPRS Receive Process state diagram

143.6.8.2 Output process

The 100G-EPON MPRS Onput Process shall implement the state diagram as depicted in Figure 143–1.

The Output Process outputs EQs to proper vMAC. In the case of overlapping envelopes from the same LLID, data from multiple channels is properly serialized. A corrupted header will lead to loss of an envelope, but no subsequent envelopes will be lost due to the error.



kramer_3a_3b_1116.pdf slide 3 or similar (SA Motion #6)

Figure 143–13—MPRS Output Process state diagram

Omitted:

logical_link_id

Value: 16 bits

This variable shall be set to the broadcast value of {0x7FFE} for the unregistered ONU MAC. Enabled OLT MACs may use any value for this variable. If the optional multicast LLID feature is supported, the OLT may use a multicast_link_id ~~along with the mode bit set to 0~~. Registered ONU MACs may use any value other than the reserved values listed in Table 76–4 or a multicast_link_id for this variable.

NOTE: once registration process is established and broadcast LLID values are agreed need to update Table 76-4 and the para above.