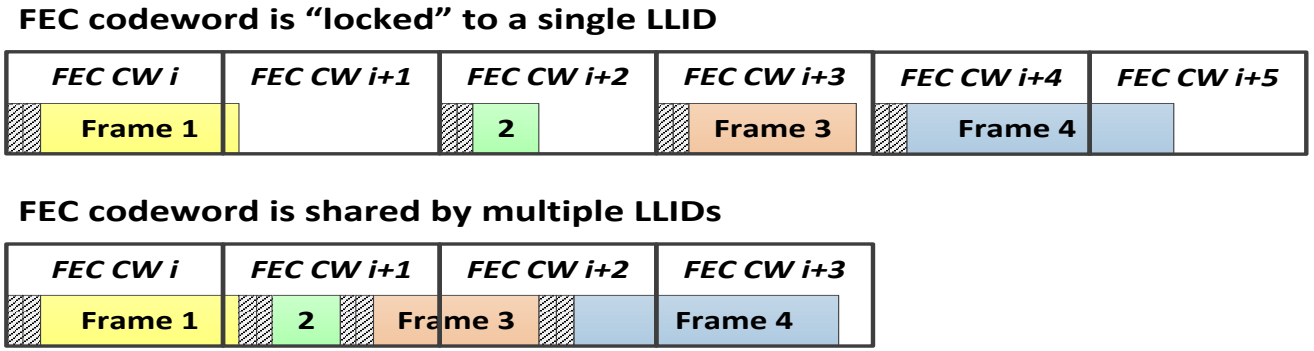
Multi-Point Reconciliation Sublayer (MPRS)
[Upstream Direction]

Glen Kramer, Broadcom
Duane Remein, Huawei
Frank Effenberger, Huawei
Mark Laubach, Broadcom
Jean-Christophe Marion, Broadcom

Motivation

- Previous Channel Bonding proposals had various shortcomings
  - **kramer_3ca_1a_0516.pdf**: Frame-based channel bonding requires very large buffers at the OLT to story up to 3 complete bursts per each LLID.
  - **kramer_3ca_3c_0716.pdf**: FEC-codeword-based channel bonding allocates an entire codeword to a single LLID. That causes an excessive overhead in upstream and downstream directions.



- Objectives of this proposal
  - Grants should be allocated to (shared among) multiple LLIDs within an ONU
  - Transmission assignments to LLIDs should have granularity finer than a whole frame or an FEC codeword (byte? Word? Dword? Qword?)
  - FEC codewords should be shareable by multiple LLIDs

MPRS Overview

Multi-Point Reconciliation Sublayer (MPRS)

- ***MPRS*** interfaces to ***M*** MAC instances above and ***L*** xMII instances below

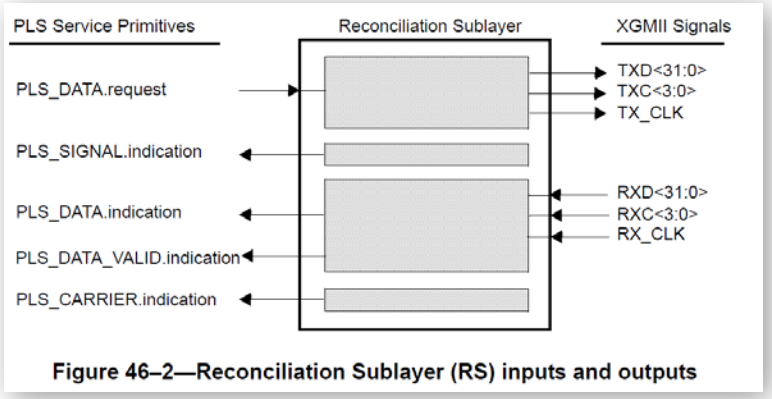
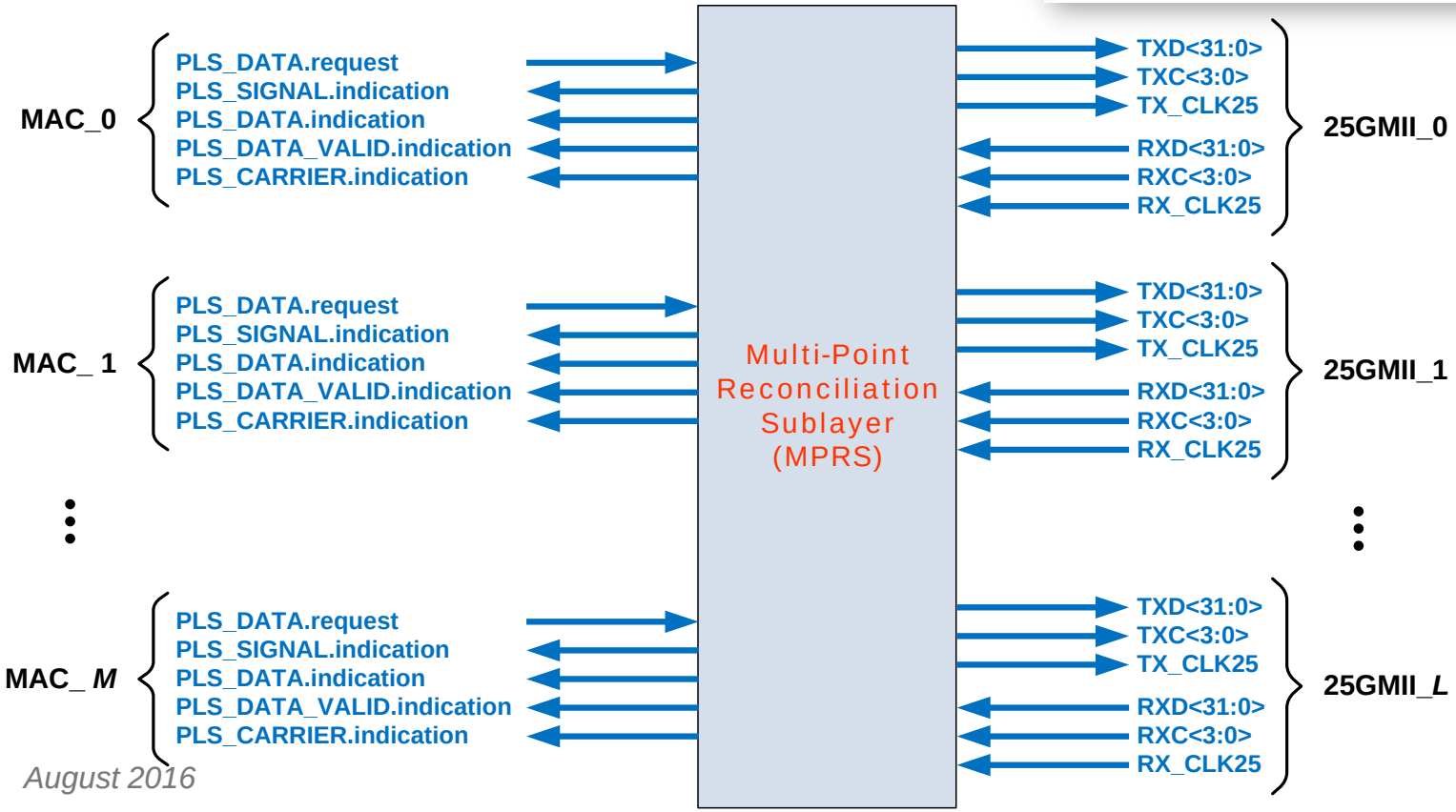


Figure 46–2—Reconciliation Sublayer (RS) inputs and outputs



Precedent for multi-interface RS

- In 802.3, we already have RS with multiple MACs above and multiple xMIIs below

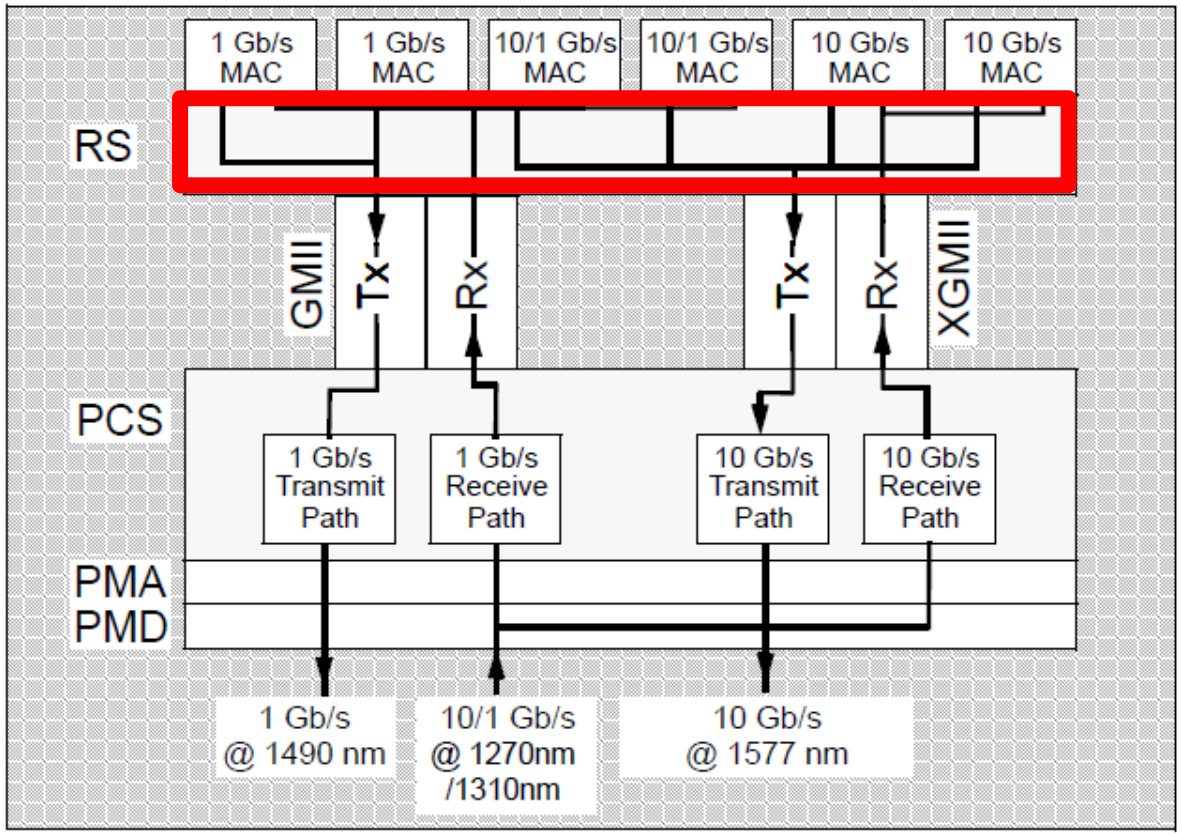
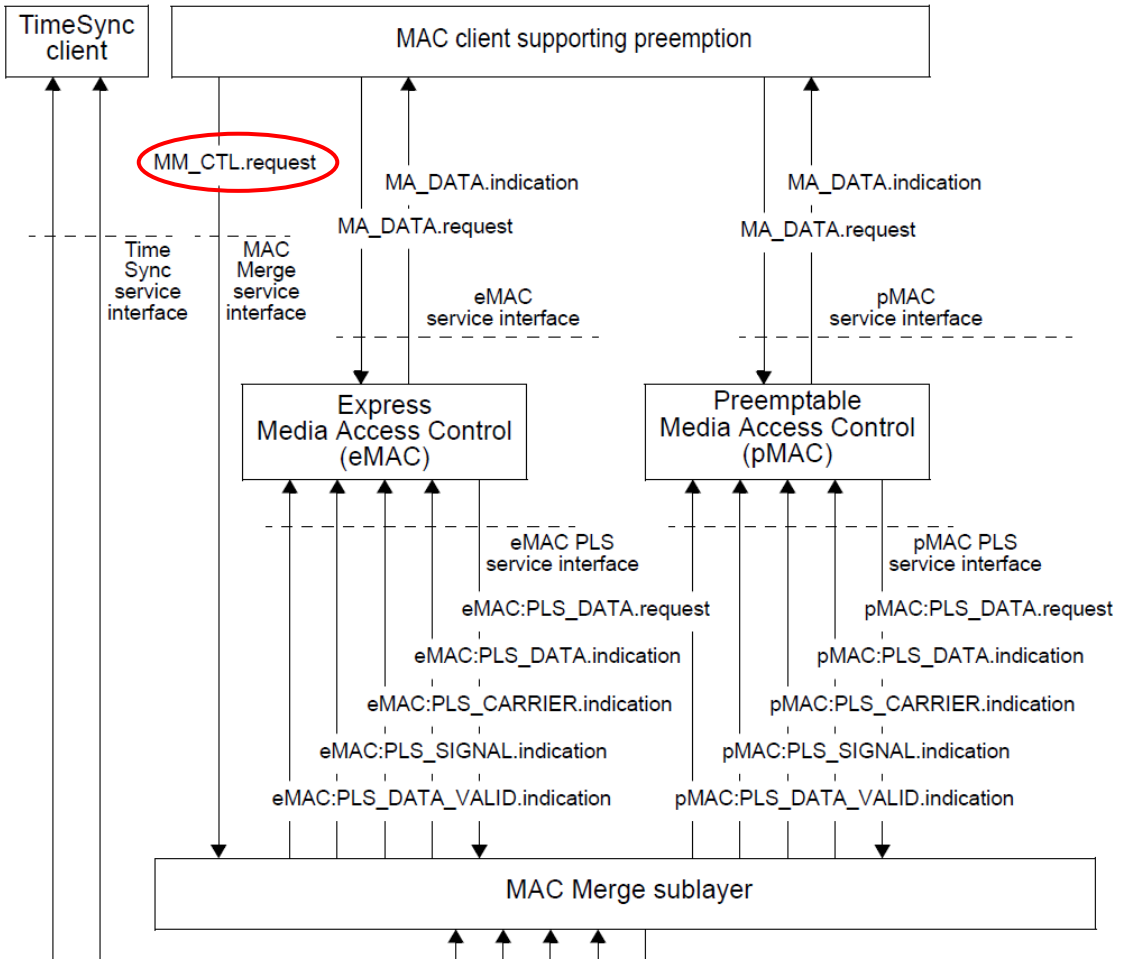


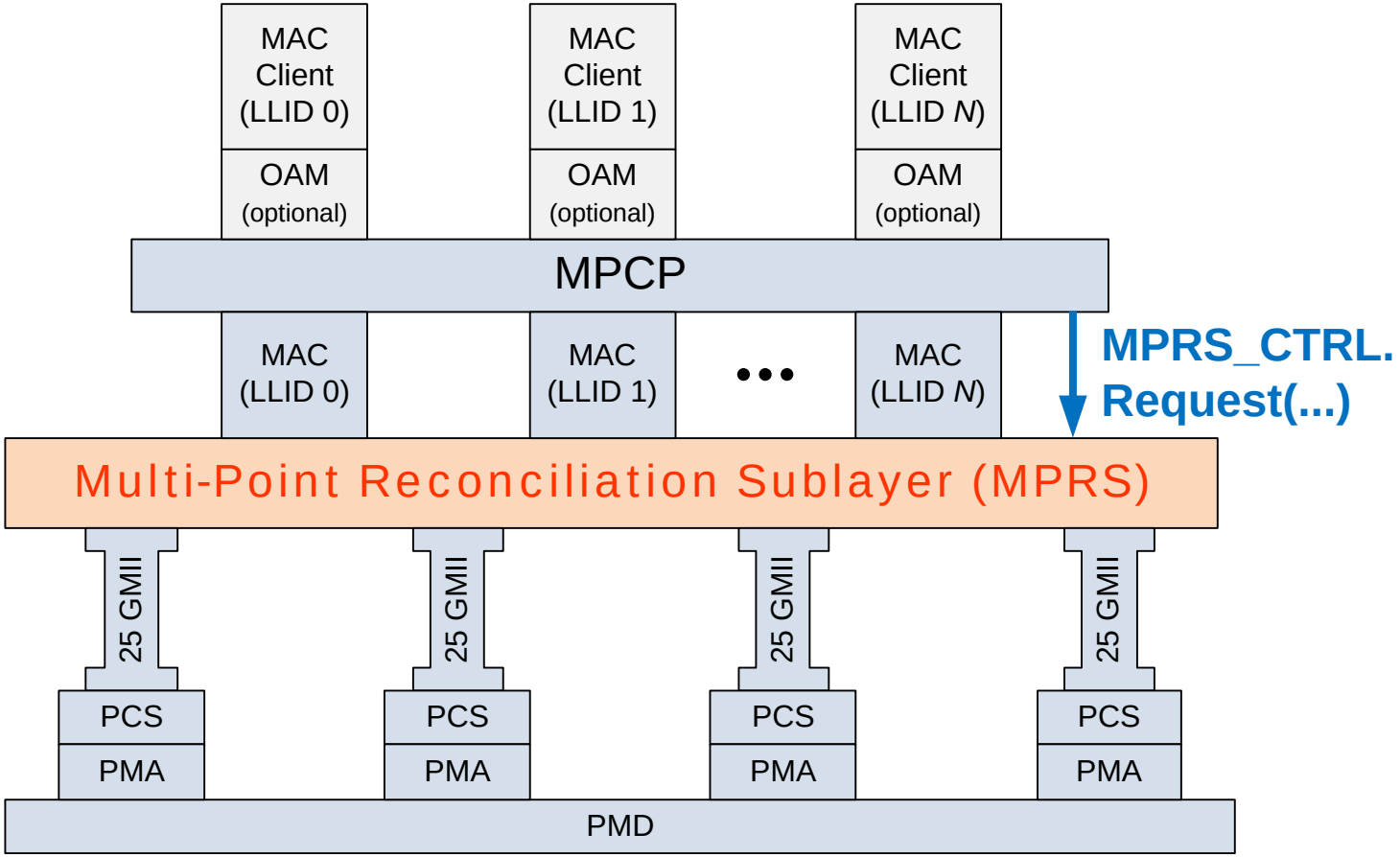
Figure 76–4—PCS and Reconciliation Sublayer for dual rate mode at OLT

MPRS Control Primitive

- **MPRS_CTRL.request()** primitive is similar to **MM_CTRL.request()** in 802.3br
- MPRS_CTRL.request() is generated in MPCP and processed in MPRS
- Detailed MPRS_CTRL.request() definition is shown later



100G ONU Layering Diagram

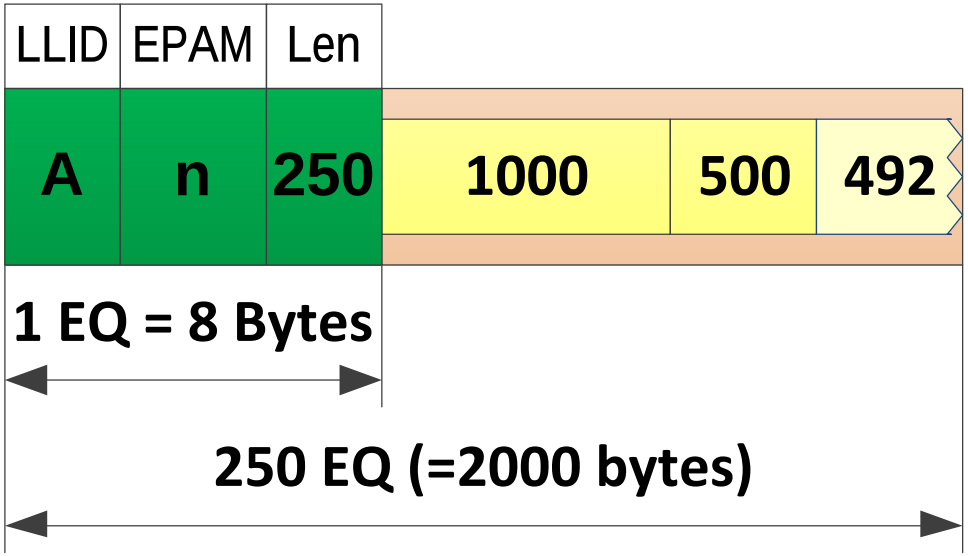


- Each MAC is a 100 Gb/s MAC
- If not all four lanes are active at a given time, the MPRS will pause MAC (i.e., not accept more bits) to equalize MAC and PHY data rates

MPRS Key Concepts

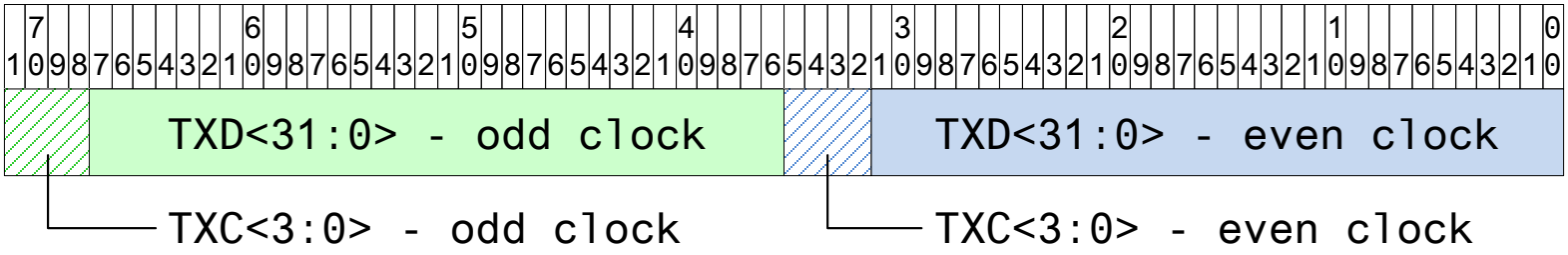
Grant Envelope

- A grant may allocate bandwidth to multiple LLIDs
- **Grant Envelope** is a structure that encapsulates a transmission by a single LLID within a grant.
- Structure of Grant Envelope:
  - Wraps multiple frames with a common header, that includes ***LLID***, ***Envelope Position Alignment Marker (EPAM)***, and ***envelope length***.
  - Envelope payload size represents the number of units (***Envelope Quanta - EQ***) granted to a given LLID in a given grant
  - Beginning of an envelope may be a full frame or a tail segment of a frame.
  - End of the envelope may be a full frame or a head segment of a frame.



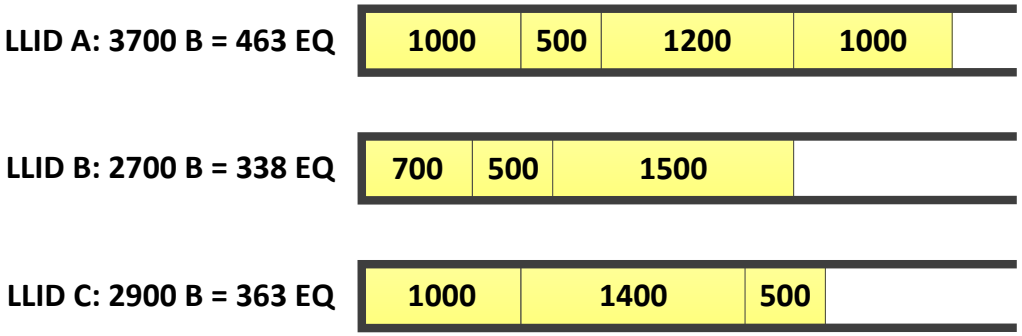
Envelope Quanta

- Envelope length is expressed in units of **Envelope Quanta (EQ)**
- EQ represents 8 bytes of data
- Within MPRS, EQ maps to two successive 25GMII transfers
- Within PCS, EQ maps to a single 64b/66b coded block



Sharing a grant among LLIDs

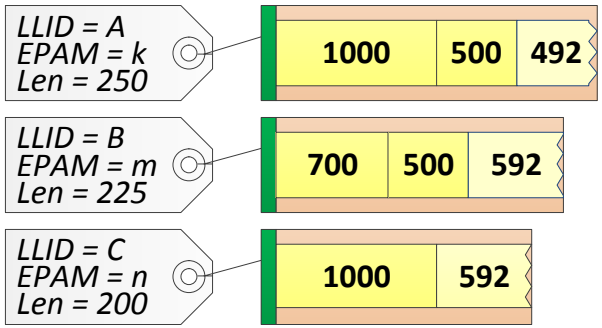
1. ONU reported this:



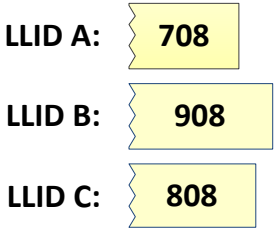
2. OLT sent this GATE2

6	DA	
6	SA	
2	Type = 0x88-08	
2	Opcode = 0x00-12	
4	Timestamp	
1	Ch = **0** / # of grnts = **3**	
4	Grant Start Time	
2	**A**	Grant item #1
4	**250 EQ (= 2000 B)**	
2	**B**	Grant item #2
4	**225 EQ (= 1800 B)**	
2	**C**	Grant item #3
4	**200 EQ (= 1600 B)**	
	Pad = 0	
4	FCS	

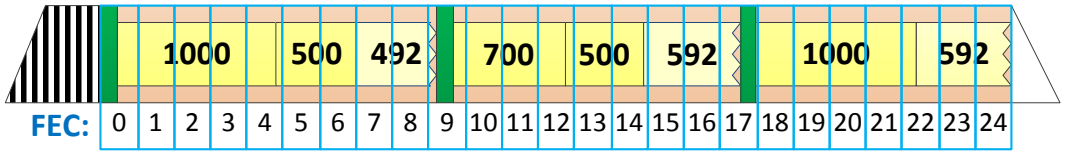
3. ONU formed these grant envelopes:



Left in MAC

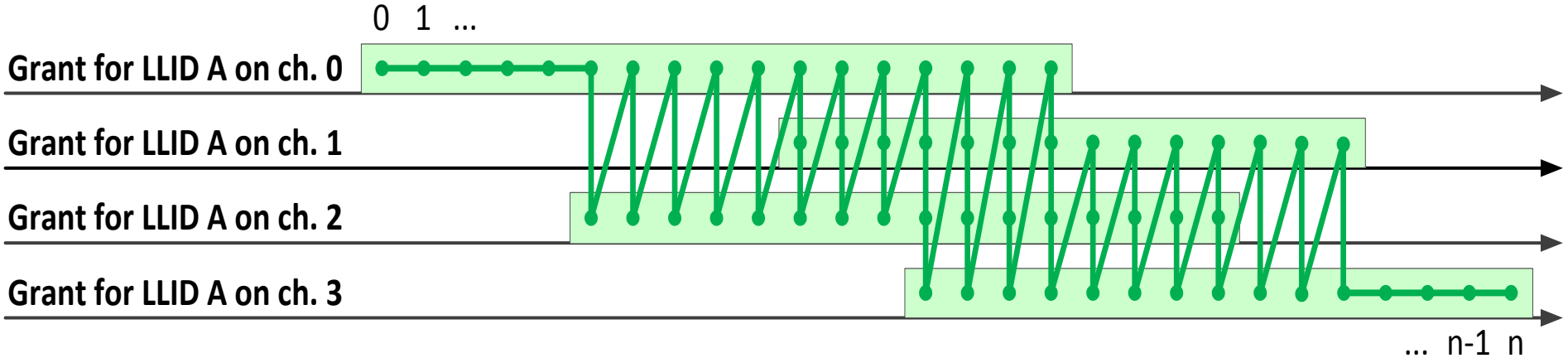


4. ONU sent this burst:



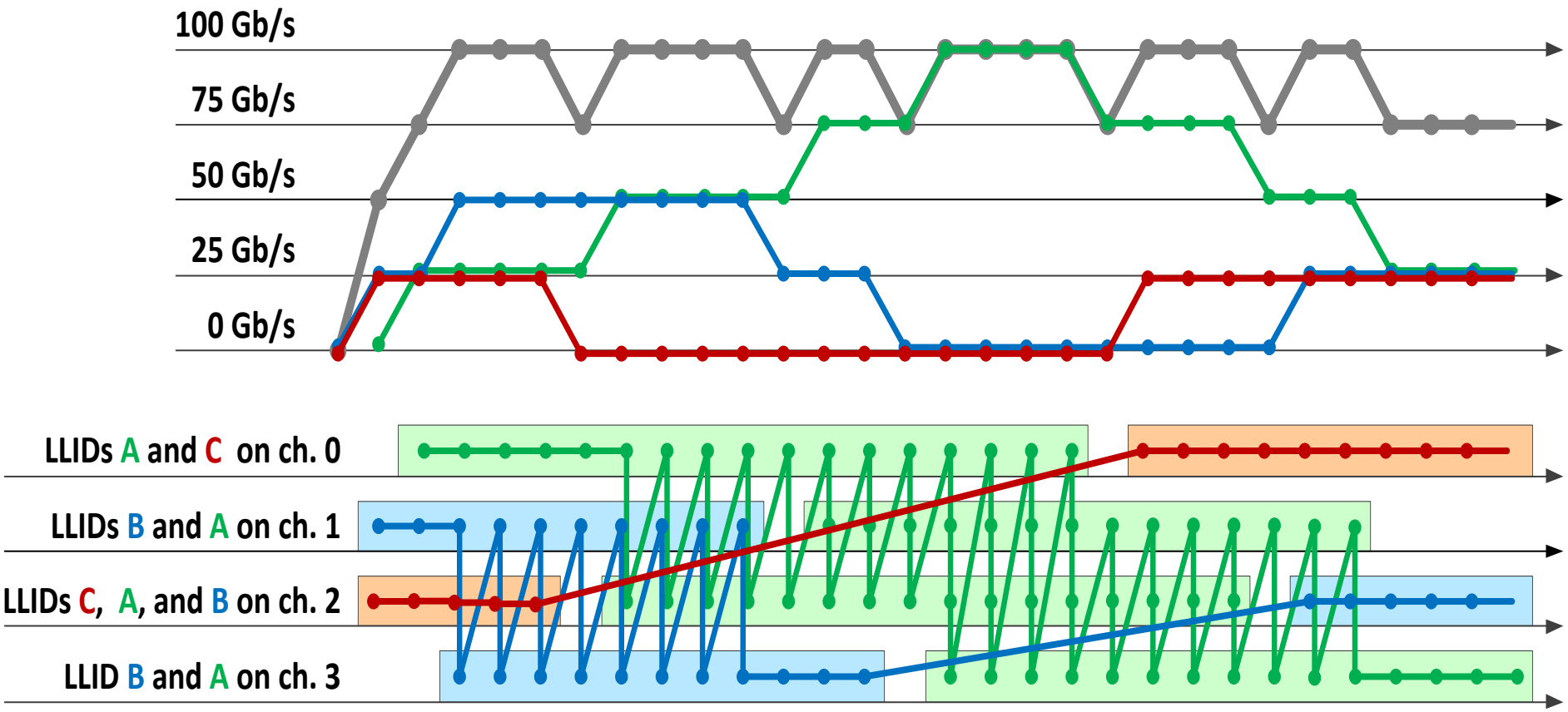
Overlapping Envelopes

- An LLID may be granted multiple overlapping grants (envelopes).
- Data is interleaved in multiple envelopes allowing for up to 100 Gb/s transmission by a single MAC (LLID)
  - Interleaving unit is EQ.



Completely flexible schedule

- Envelopes for different LLIDs can be scheduled independently on different lanes
- MPRS will automatically interleave the EQs to fully utilize granted envelopes.

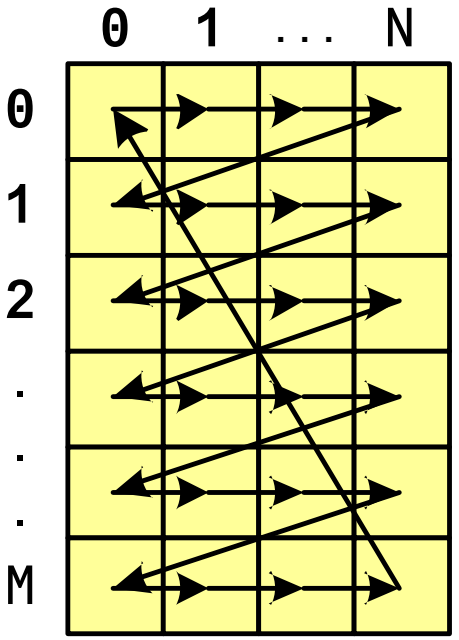


MPRS Operation

2D Alignment Buffer

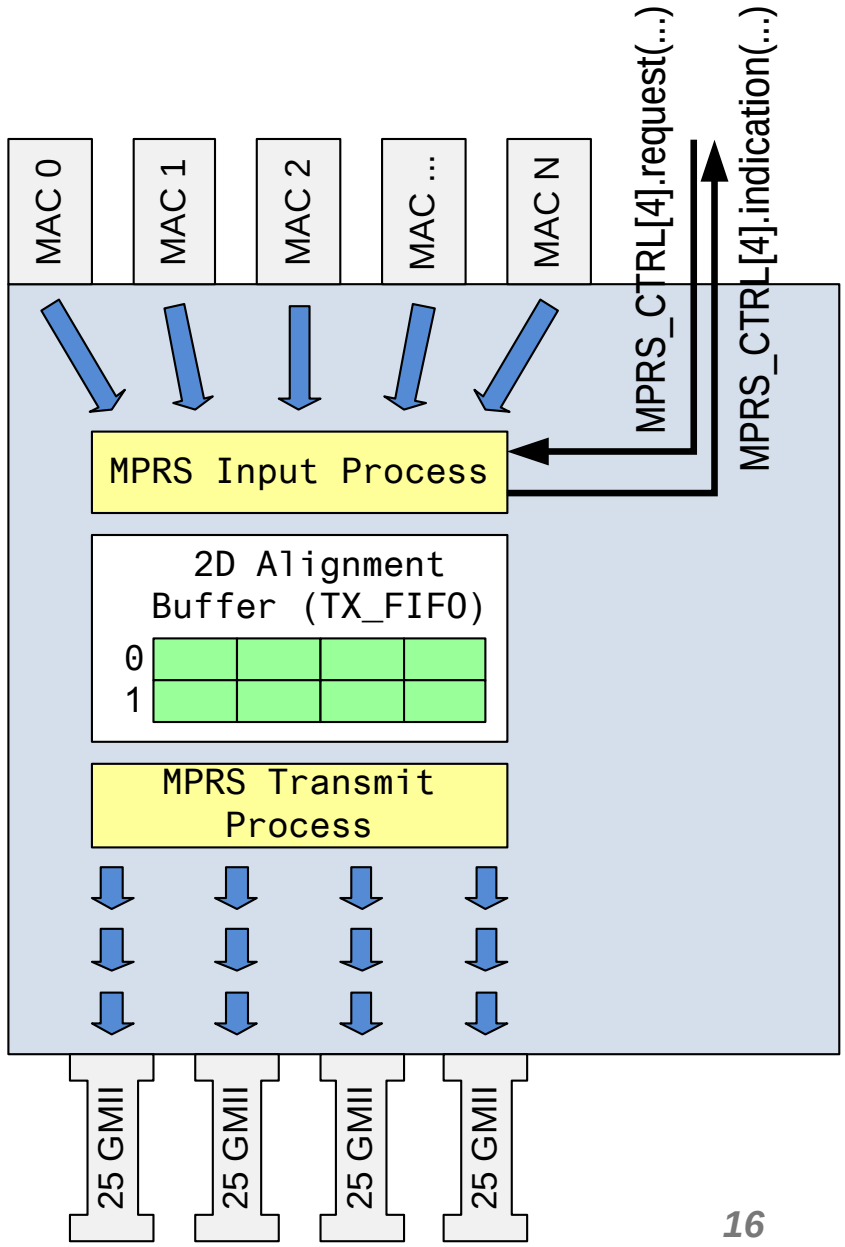
- The key part of MPRS is the 2D alignment buffer, called **TX_FIFO** in the ONU (and **RX_FIFO** in the OLT)
  - Each cell in the buffer stores one EQ (EQ_raw, which is a 72-bit vector)
  - The buffer has N columns and M rows.
    - ***N*** – number of channels
      - ***N***=4 for 100 Gb/s ONU
      - ***N***=2 for 50 Gb/s ONU
      - ***N***=1 for 25 Gb/s ONU
    - ***M*** should be twice as large as the maximum upstream propagation delay variability (MPRS@ONU to MPRS@OLT)
      - **Note:** Because TX side does not need to compensate for the skew, in TX_FIFO ***M*** can be reduced to 2
  - The buffer is filled in cyclic pattern row-by-row
  - The source LLID for each cell is determined by the grants

TX_FIFO in ONU
RX_FIFO in OLT



Main blocks in ONU MPRS Tx Path

- **MPRS Input Process**
  - Accepts data from MAC interfaces into TX_FIFO one EQ at a time
  - Prepends a header to each envelope
- **MPRS Transmit Process**
  - Outputs one 36-bit vector (TXD<31:0>+TXC<3:0> to each 25GMII interface one every clock TX_CLK
- Only one instance of each process is running in a physical device



Transmission operation

1. Data waiting in a queue (each box = 1 EQ)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

2. Scheduled envelopes for LLID A

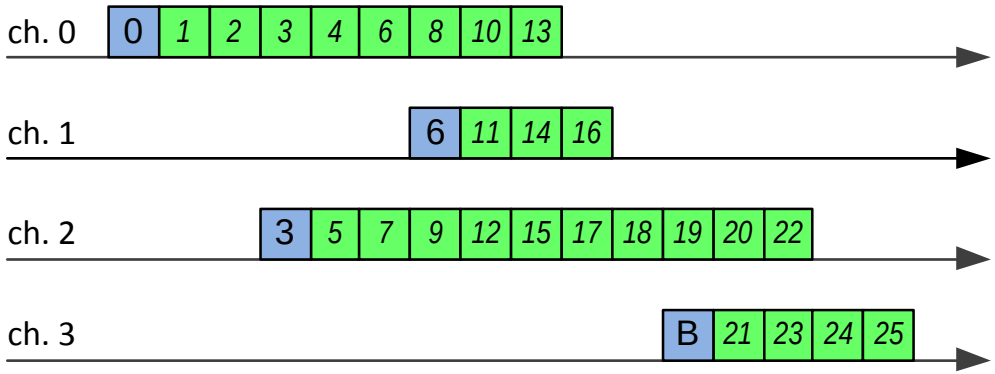
1) on channel 0: Start=T_0; Length = 9 EQ
2) on channel 1: Start=T_1; Length = 4 EQ
3) on channel 2: Start=T_2; Length = 11 EQ
4) on channel 3: Start=T_3; Length = 5 EQ

3. TX_FIFO Queue

(Blue boxes = envelope headers)

T_0 -	0	0		
	1	1		
	2	2		
T_2 -	3	3		3	
	4	4		5	
	5	6		7	
T_1 -	6	8	6	9	
	7	10	11	12	
	8	13	14	15	
	9		16	17	
	A			18	
T_3 -	B			19	B
	C			20	21
	D			22	23
	E				24
	F				25

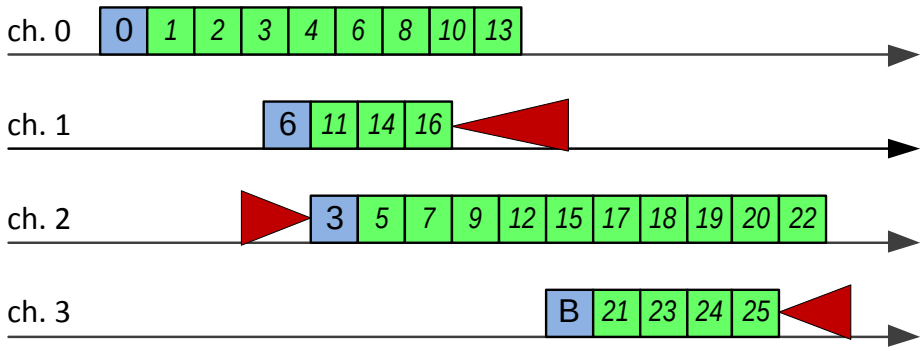
4. Transmitted envelopes for LLID A



Reception operation

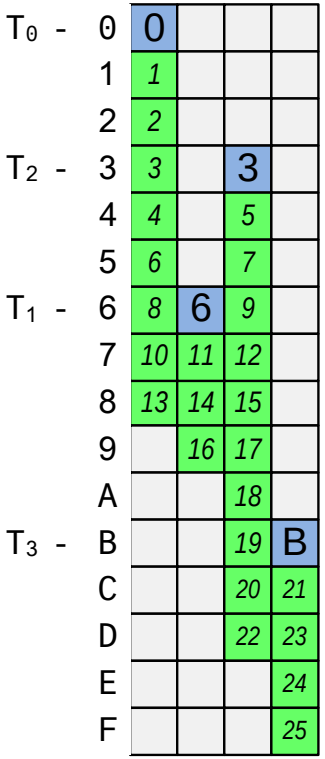
1. Received envelopes for LLID A

(Red triangles show skew magnitude and direction)

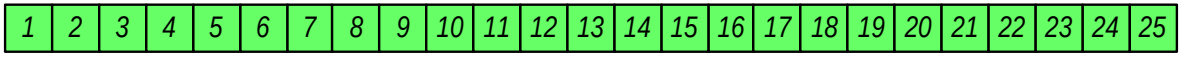


2. RX_FIFO Queue

(Blue boxes = envelope headers)



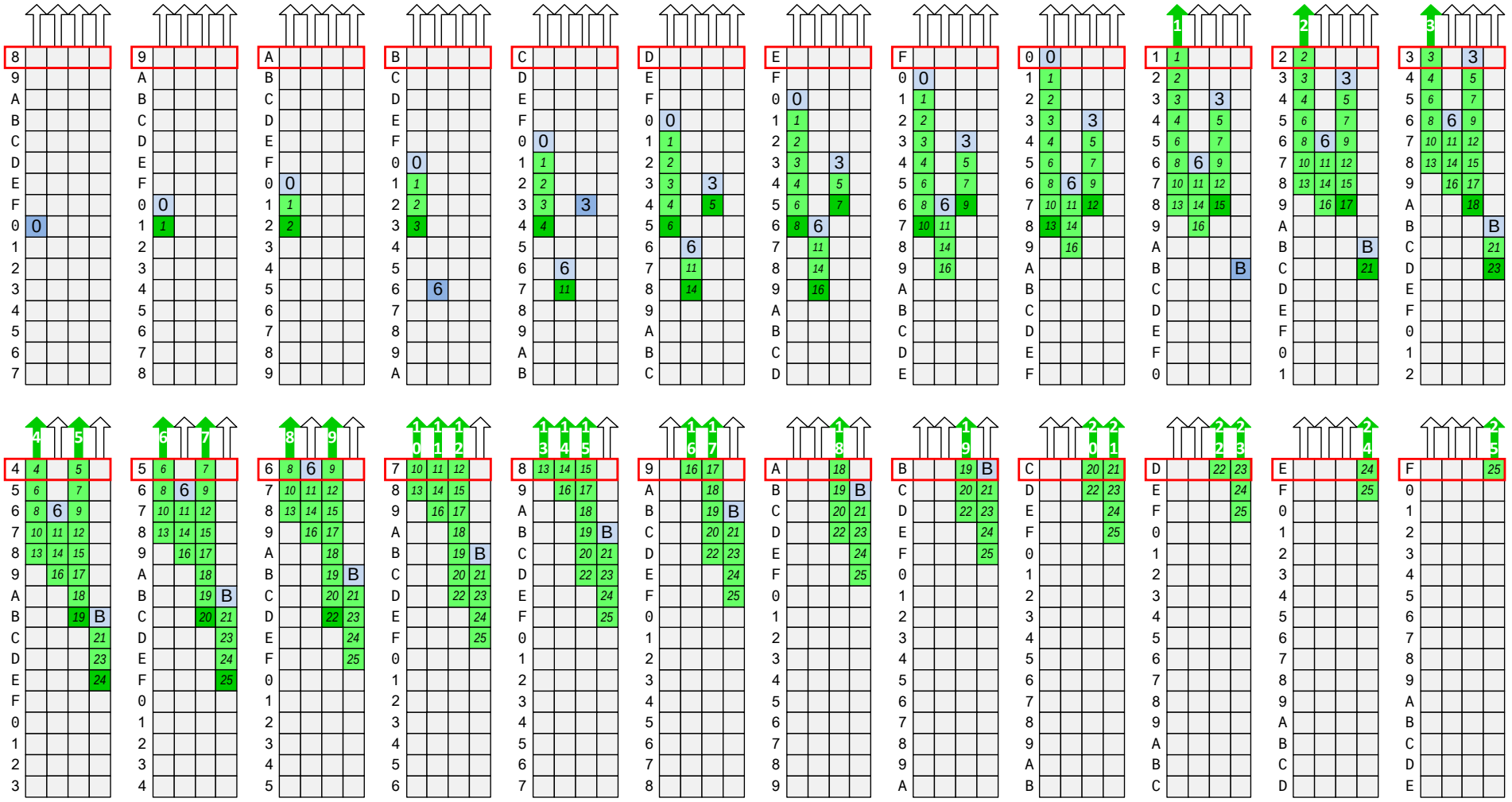
3. Data passed to MAC (each box = 1 EQ)



- RX_FIFO is filled not based on data arrival times, but based on the Envelope Position Alignment Marker (EPAM) carried in each envelope header.
- RX_FIFO alignment looks identical to TX_FIFO alignment
- EQs are passed to the receiving MAC in the same order that they were taken from the transmitting MAC

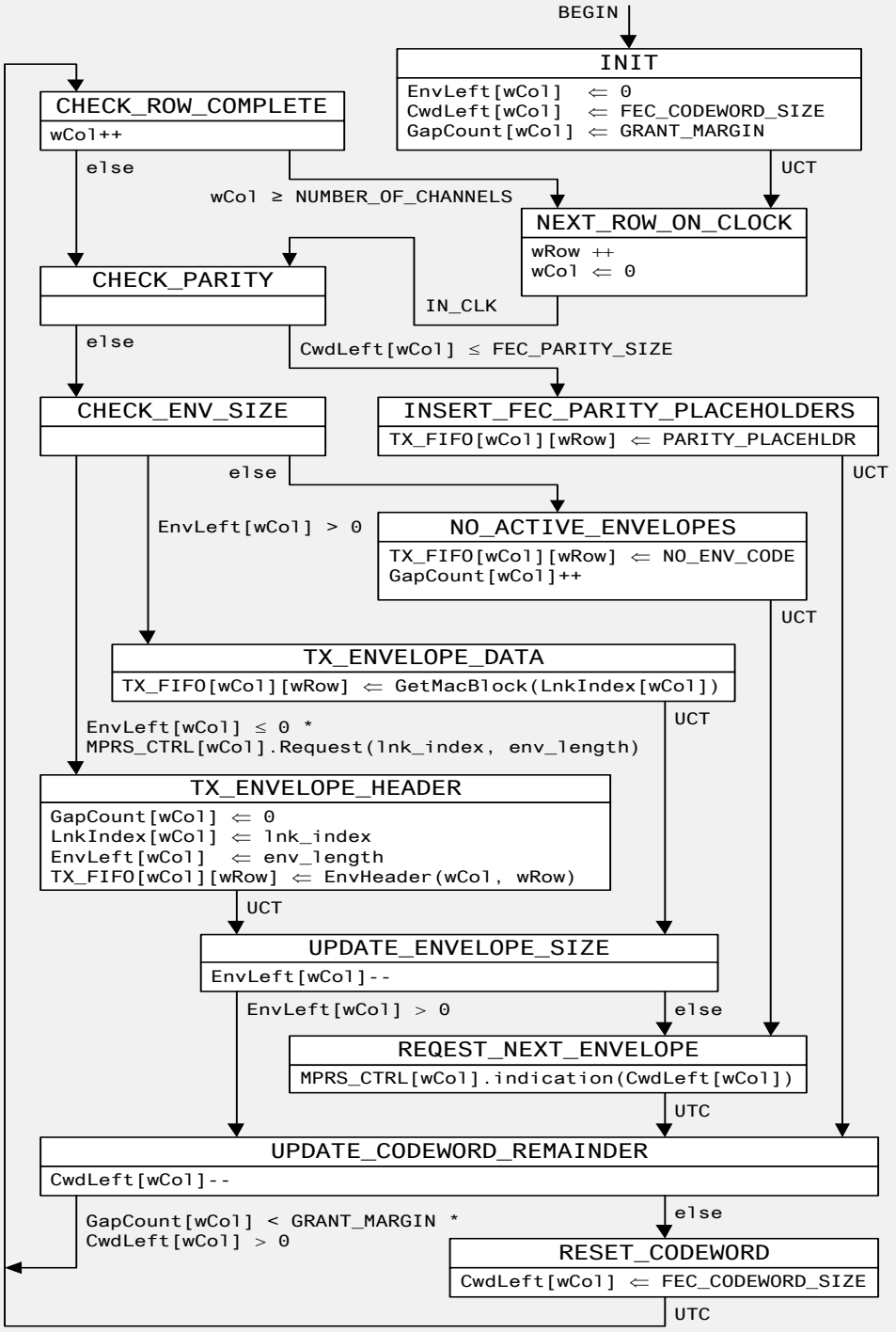
RX_FIFO Time Spread

- Write position - Read position = ***M / 2***
- RX_FIFO introduces a base delay ***D = M / 2 * IN_CLK***
- Actual delay is channel-dependent: $D_{CH} = D + Skew_{CH}$ (skew can be positive or negative)



Input Process

- Input process fills one row in TX_FIFO buffer on each tick of input clock IN_CLK (half of TX_CLK)
  - In case of overlapping envelopes, blocks in multiple columns will be retrieved from the same MAC
- Input process keeps track of the envelope sizes for each LLID and won't exceed the allowed number of EQs.
- Input process adjusts MAC rate to account for FEC parity insertion in the PCS.



EQ types

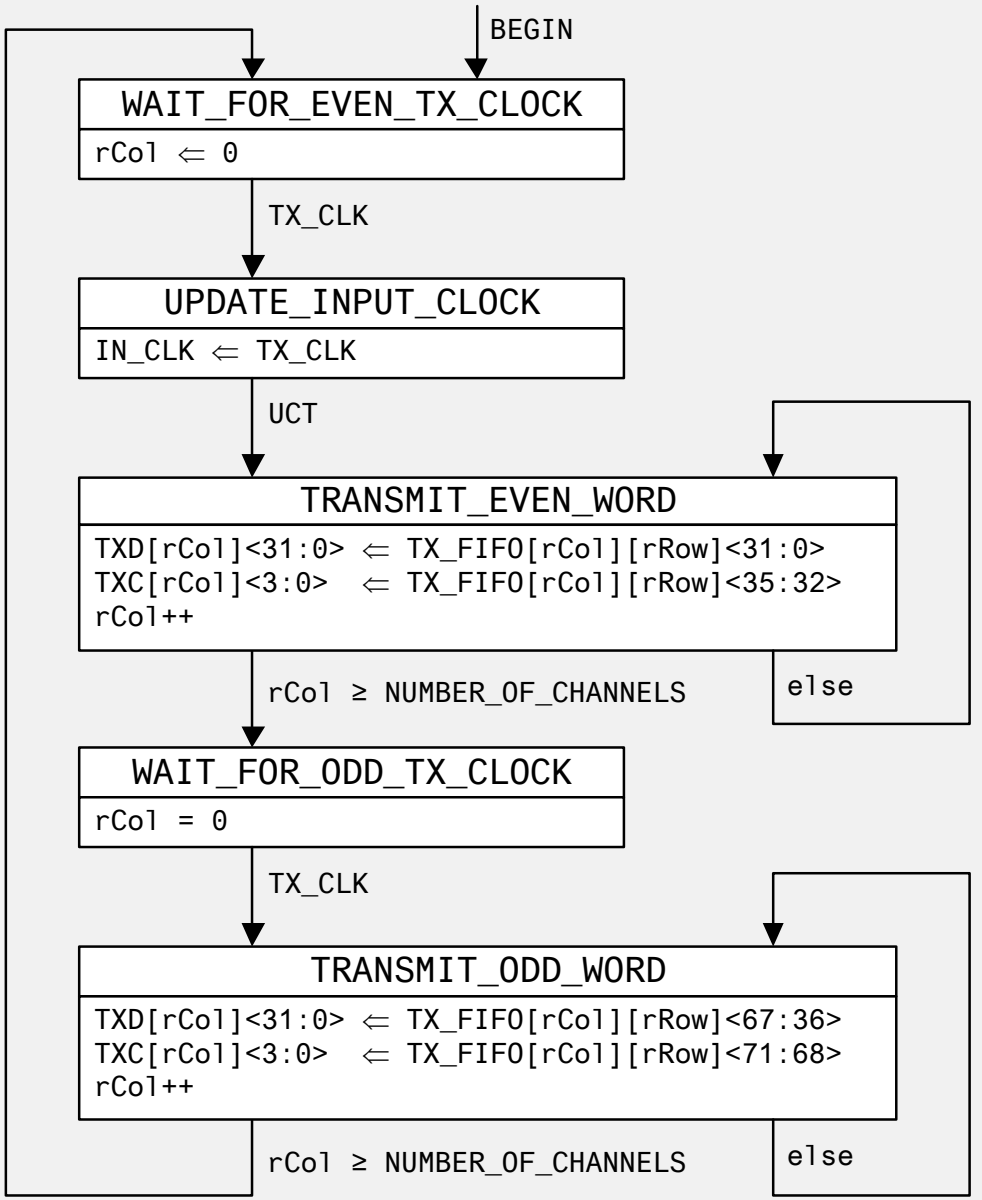
- On each main loop through the Input process, all columns in one row are filled.
- For each TX_FIFO cell, the Input process selects one of 4 types of EQs:
  1. **PARITY_PLACEHLDR** – special control code written when data from MAC should be paused to allow FEC parity insertion. In the PCS sublayer, PARITY_PLACEHLDR code is overwritten by the calculated parity values
  2. **NO_ENV_CODE** – special control code written when no active envelope exists (i.e., the bandwidth was not assigned to any LLID).
  3. **Data EQ** – an EQ that contains data (or idles) retrieved from the MAC
  4. **Envelope Header EQ** – an EQ containing LLID value, Envelope Position Alignment Marker (EPAM), and envelope length. Other fields are TBD.

MMRS_CTRL Primitives

- The Input process requests the next envelope from MPCP after the completion of the previous envelope using **MPRS_CTRL[ch_index].indication(…)** primitive
  - ***MPRS_CTRL[ch_index].indication(cw_left)*** indicates to the MPCP that 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_index].indication() 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 a certain margin (GRANT_MARGIN) sufficient to turn on the laser, every MPRS_CTRL[ch_index].indication() 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.
- The MPCP requests the MPRS to transmit the next envelope using the **MPRS_CTRL.request(…)** primitive
  - ***MPRS_CTRL[ch_index].request(lnk_index, env_length)*** opens an envelope on a channel *ch_index* for LLID *lnk_index* of length *env_length* EQs

Transmit Process

- Main function – transmit full row from TX_FIFO buffer on all existing channels
- Transmit process is synchronized on TX_CLK
- One iteration of transmit process takes two clocks, with one 25GMII transfer taking place on each clock
- Input clock is updated on even ticks of TX_CLK and runs at half the rate



State Diagram Variables (1/2)

CwdLeft - This variable counts the remaining space in the FEC codeword. 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 - Number of EQs that remain to be transmitted in the current envelope. A separate instance of this variable exists for each channel.

GAPCount – This variable counts the number of continuous NO_ENV_CODE EQ values. This value exceeding the GRANT_MARGIN indicates that the laser is turned off and the next envelope will begin a new burst.

IN_CLK - Clock signal that corresponds to even transitions of TX_CLK. Therefore IN_CLK runs at half the frequency of TX_CLK.

LnkIndex - Index of the LLID that is sourcing data for the current envelope. A separate instance of this variable exists for each channel. Different envelopes on different channels may source data from the same LLID.

State Diagram Variables (2/2)

rCol - An integer that represents the column in TX_FIFO buffer currently being read. Each column corresponds to a separate transmission channel, i.e., a separate 25GMII interface.

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

TXD<31:0> - See definition of 25GMII in 802.3by

TXC<3:0> - See definition of 25GMII in 802.3by

wCol - An integer that represents the column in TX_FIFO buffer currently being populated. Each column corresponds to a separate transmission channel, i.e., a separate 25GMII interface.

wRow - 4-bit integer that represents the row in TX_FIFO buffer currently being populated. This variable also represents the envelope position alignment marker and is written in the envelope header.

State Diagram Functions (1/1)

EnvHeader (LnkIndex, EnvLength, AlignmentPos) –

This function returns an EQ representing an envelope header, which has the following format:

Bits	Value	Description
0-15	varies	LLID value for the given envelope
16-19	varies	Envelope Position Alignment Marker (Number of bits matches the size of wRow)
20-31	0x000	Reserved (pad)
32-35	0x0	Control bits corresponding to TXC<3:0>
36-59	varies	Length of envelope (in EQ)
60-67	0x00	Reserved (pad)
68-71	0x0	Control bits corresponding to TXC<3:0>

GetMACBlock(LnkIndex) -

This functions retrieves 8 octets (64 bits) of data from a MAC identified by LnkIndex parameter. The function returns an EQ that contains both the data and the corresponding 8 control bits. This is a blocking function that returns the control to the calling routine after 64 successive invocations of PLS_DATA.request() primitive.

State Diagram Primitives (1 / 1)

MPRS_CTRL[ch_index].indication(cw_left) -

This primitive indicates to the MPCP that MPRS is ready to start transmission of a next envelope.

Parameters:

cw_count – the number of EQs left in the current FEC codeword at the moment when the primitive is generated. This parameter allows the MPCP to control whether to start the next envelope immediately after the previous envelope ended (back-to-back) or to align the next envelope with the beginning of a FEC codeword.

MPRS_CTRL[ch_index].request(lnk_index, env_length) -

This primitive specifies an envelope parameters to be transmitted next. The MPRS will start the transmission of this envelope on the next IN_CLK after the reception of this primitive.

Parameters:

lnk_index – index of the LLID that is to source the data

env_length – length of the envelope, including the header, in units of EQ.

Thank You