Updates to ONU MPRS state diagrams

Glen Kramer, Broadcom
Jean-Christophe Marion, Broadcom
Duane Remein, Huawei

Introduction

- The ONU MPRS state diagrams were accepted as a baseline at the September meeting in Fort Worth:

Motion #5
Adopt the ONU upstream channel bonding proposal as presented in kramer_3ca_2b_0916.pdf and develop a similar mechanism for the downstream. This motion supersedes Motion #4 from July 2016 meeting.

Moved: Glen Kramer Seconded: Duane Remein

For: 20 Against: 0 Abstain: 4

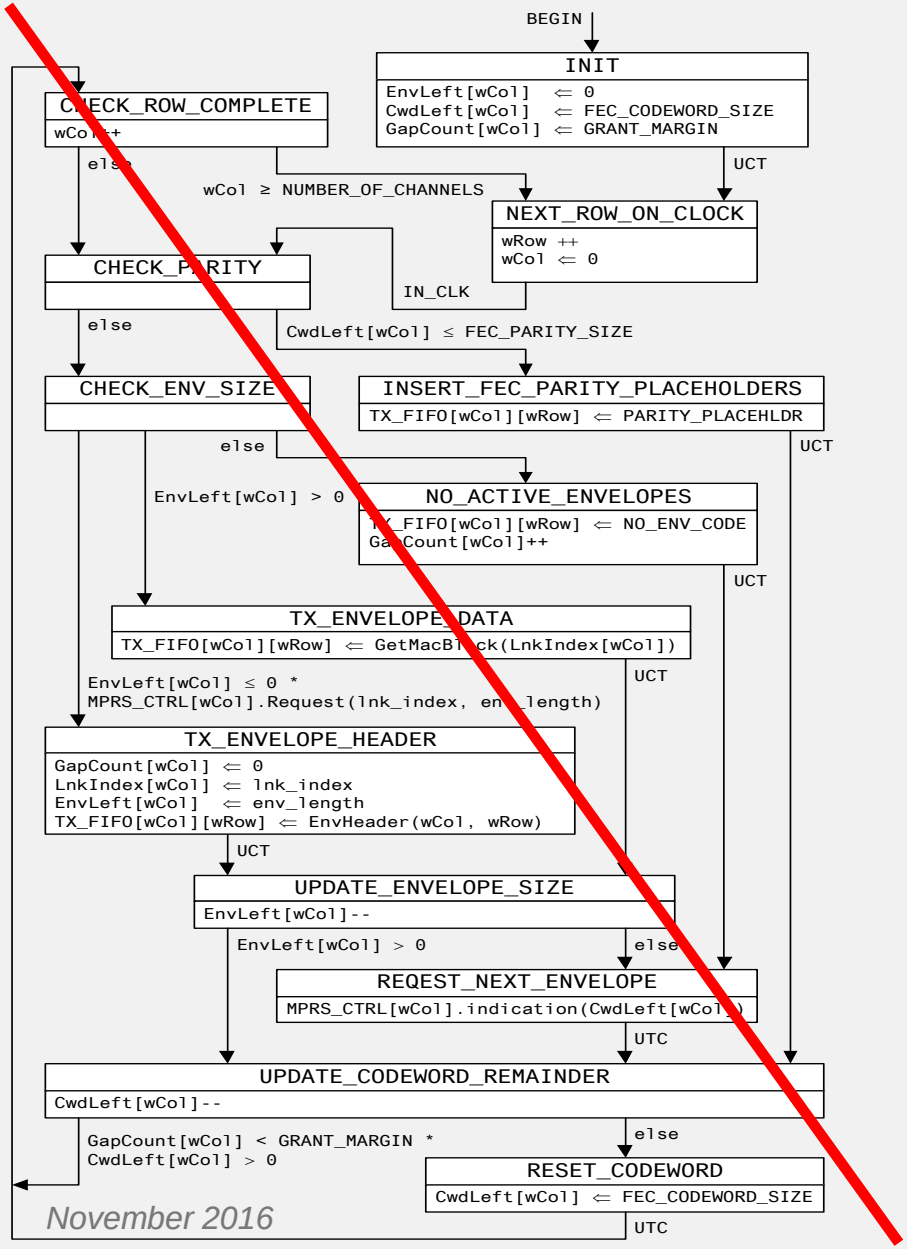
Technical ≥ 75% Passed

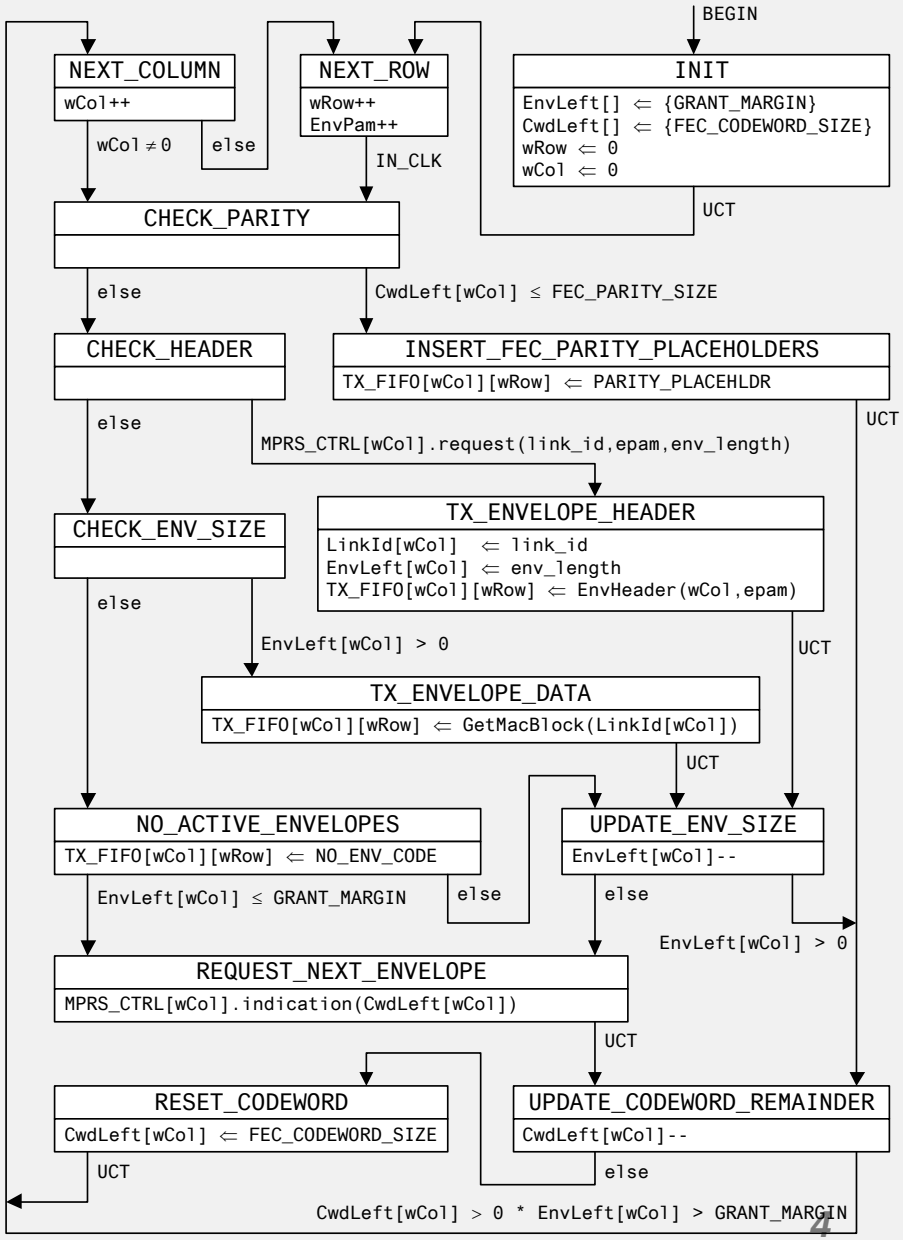
RECORDER'S NOTE: after the close of the meeting it was noticed that motion #5 refers to a non-existent file kramer_3ca_2b_0916.pdf. The correct file name should have been kramer_3ca_2c_0916.pdf.

- Good news: The MPRS behavior is stable and no major issues were found.
- A few optimizations and minor bug fixes are suggested below.

ONU MPRS Input Process

Old and New State Diagrams





#1: Initialization

- The INIT state should initialize variables for all channels, not just a single channel.

Changes:

1. Changed initializations for ***EnvLeft[]*** and ***CwdLeft[]*** to initialize entire arrays

2. Added initialization for ***wRow*** and ***wCol***

#2: Channel increment

- ***wCol*** variable represents a column in TX_FIFO. It also maps directly to a channel.
- ***wCol*** is a 2-bit integer, which wraps around after value 3.
- Check ***wCol ≥ NUMBER_OF_CHANNELS*** is invalid. Instead, the SD should simply move to the next row when ***wCol = 0***

#3: New Header Detection

- Old behavior:
  - A new envelope header is detected only when a given channel is free, i.e., if ***EnvLeft[ch] = 0***.
- New behavior:
  - A new envelope header is detected even if previous envelope is still being transmitted.
  - When a new header is detected, current envelope (if any) is terminated.
- Changes:
  - Check for new header is moved ahead of the check for remaining envelope size (see transitions CHECK_PARITY → CHECK_HEADER → TX_ENVELOPE_HEADER)

#4: GapCount variable

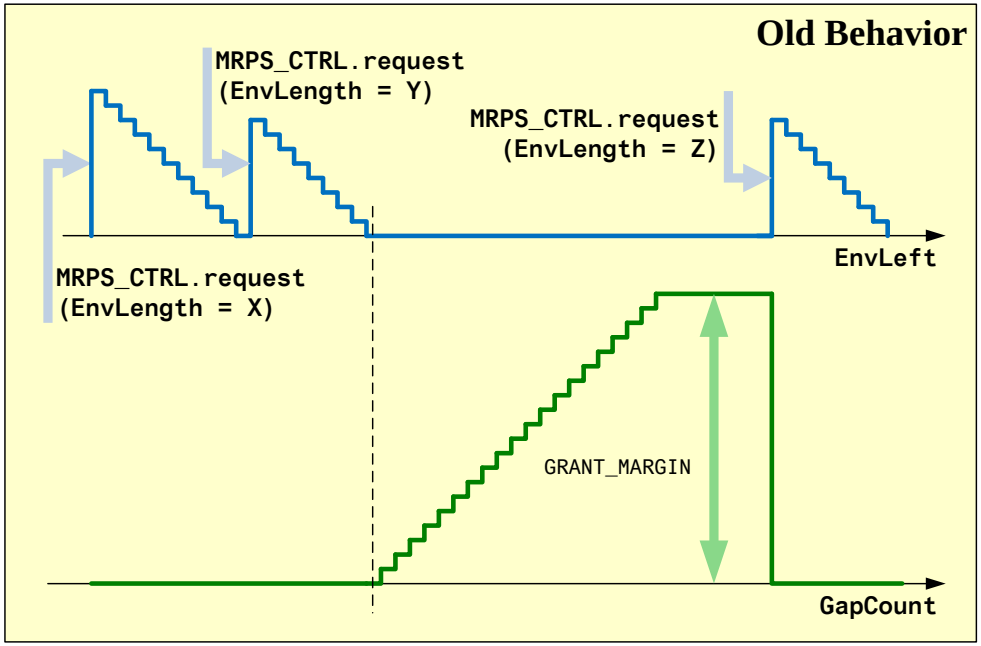
- ***GapCount*** variable is redundant. It is incremented only when ***EnvLeft*** is zero. Similarly, ***EnvLeft*** is incremented only when ***GapCount*** is zero.

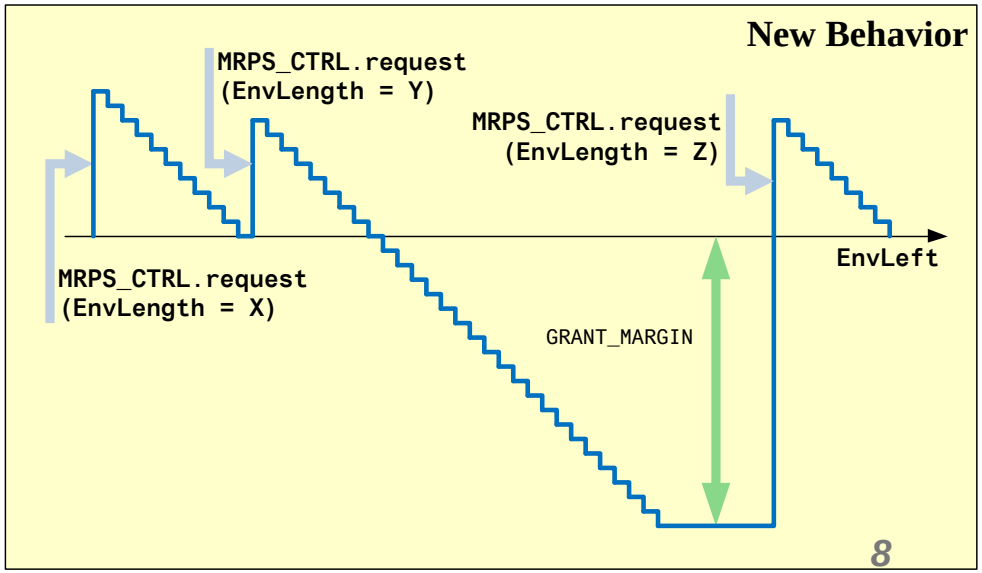
- Changes:

  1. ***GapCount*** variable is eliminated

  2. ***EnvLeft*** variable is defined as signed integer. Positive values represent the remaining envelope length (in EQ). Negative values represent the number of EQs since the end of the last envelope on a given channel.

  3. Added check for ***EnvLeft*** overflow (see transition NO_ACTIVE_ENVELOPES → UPDATE_ENV_SIZE)





#5: EPAM Insertion

- To facilitate alignment of received envelopes at the OLT, the envelope header carries Envelope Position Alignment Marker (EPAM).
- The alignment mechanism is explained in kramer_3ca_3a_1116.pdf.

- Changes:

1. EPAM is passed to MPRS Input Process from the MPCP
2. ***EnvHeader*** function uses the received EPAM value for the first envelope in a burst.

```
EQ 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;
 }
```

#6: TX_FIFO Buffer Size

- It was previously mentioned (see kramer_3ca_2c_0916.pdf) that it would suffice for the TX_FIFO buffer at the ONU to have only two rows
  - Total TX_FIFO size = 2 rows x 4 column x 72-bits = 576 bits

Changes:

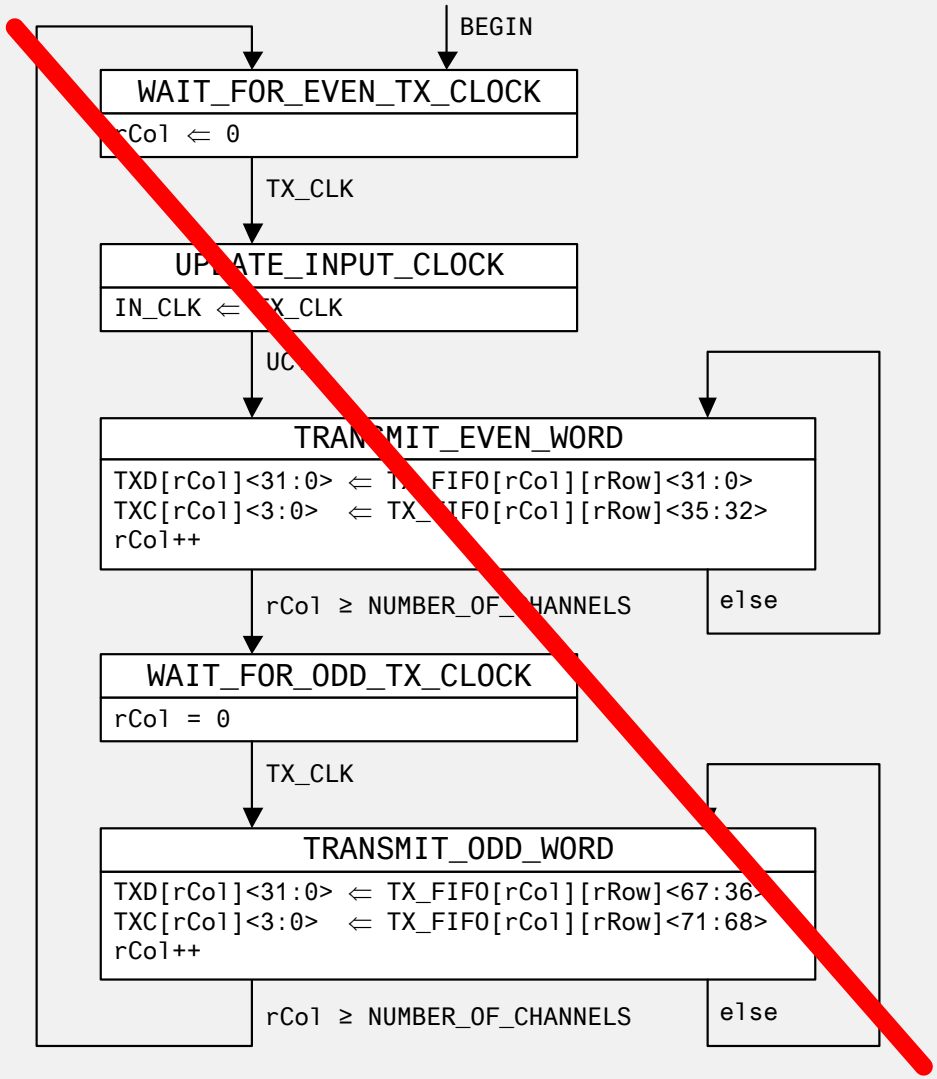
1. ***wRow*** variable is separated from the ***EnvPam*** variable.
2. ***wRow*** is a 1-bit integer that represents row index in the ***TX_FIFO*** in the ONU.
3. ***EnvPam*** is 5-bit integer that represents row index in the ***RX_FIFO*** at the OLT.

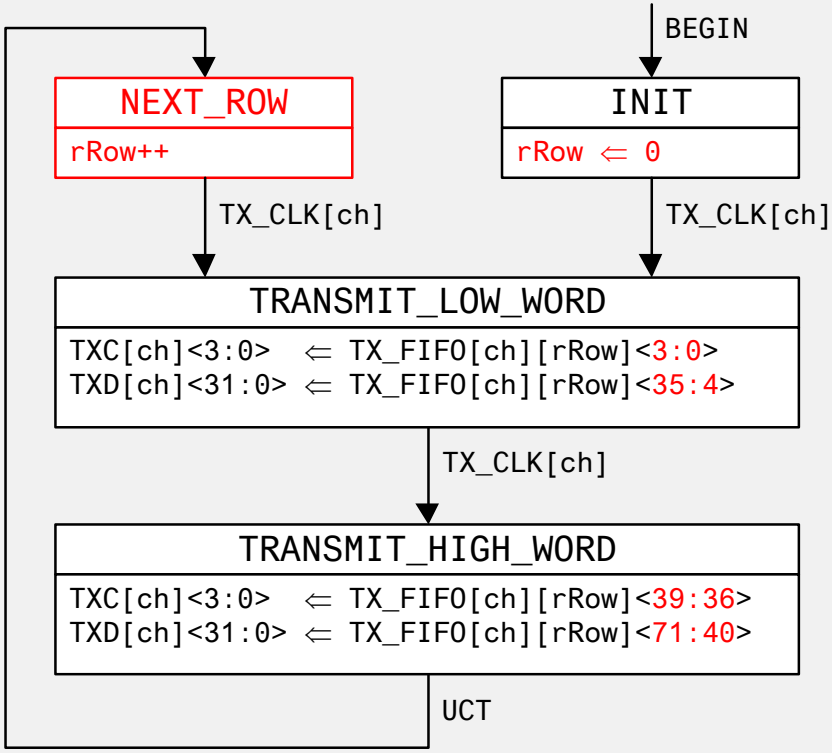
#7: Link Index

1. Rather than operating with link indices, the state diagram is changed to operate with link IDs (i.e., LLIDs).
   - This reduces the number of look-ups.
   - The ***LinkId*** value is converted into link index only once, inside the function ***GetMacBlock(…)***

ONU MPRS Transmit Process

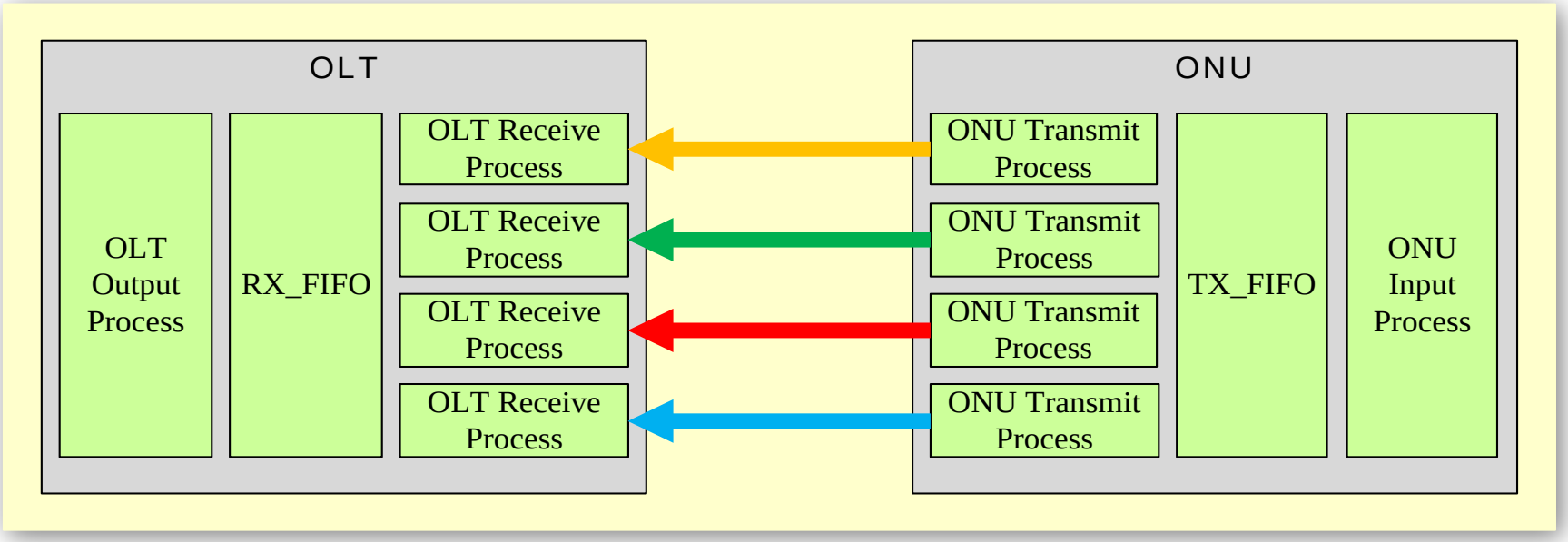
Old and New State Diagrams





#1: Separate SD Instance per Channel

- Old Behavior:
  - One instance of state diagram handled all four channels.
- New Behavior:
  - The SD is changed to operate as a single instance per channel, to make it symmetric with the OLT Receive Process.
    (The OLT must have separate instances of the Receive Process because the RX_CLK on every channel has a different phase.)



#2: TX_FIFO Buffer Size

- ***rRow*** should be defined similarly to ***wRow*** (1-bit integer)

Changes:

1. Added initialization for ***rRow***
2. Added state NEXT_ROW to increment ***rRow*** after every other 25GMII transfer.

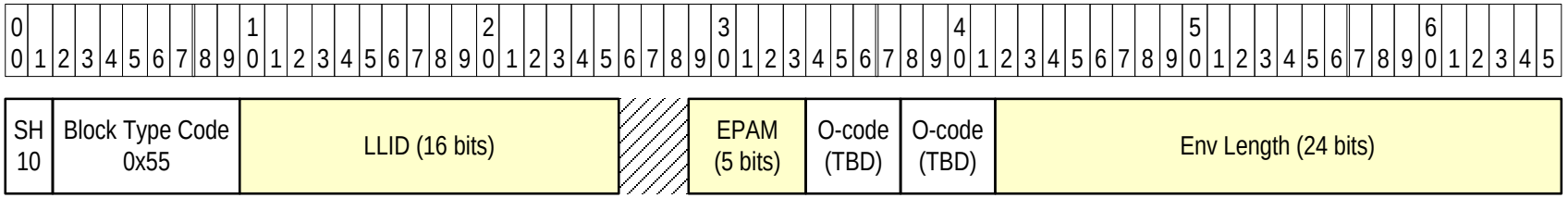
- Note that while the ***wRow*** is initialized to 1, the ***rRow*** is initialized to 0. Both variables are incremented synchronously. Thus, while the Input Process writes into row 1, the Transmit Process reads row 0. Then, when the Input Process writes into row 0, the Transmit Process reads from row 1.

#3: TX_FIFO Buffer Size

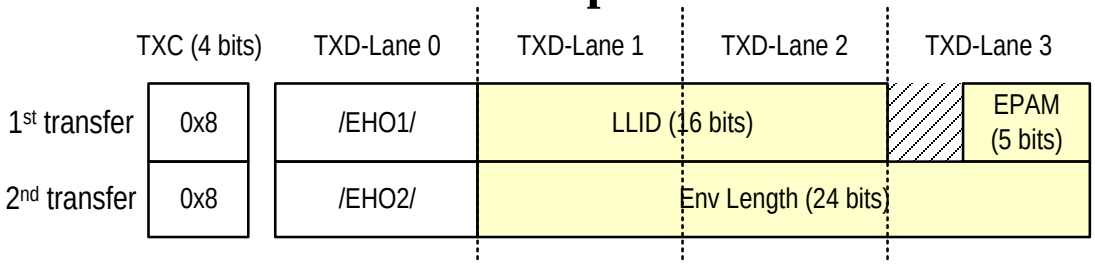
- Bit positions in states TRANSMIT_EVEN_WORD and TRANSMIT_ODD_WORD are corrected to match the EQ format (see the next slide)

Envelope Header Format

66-bit encoding of Envelope Header



25GMII transfer of Envelope Header



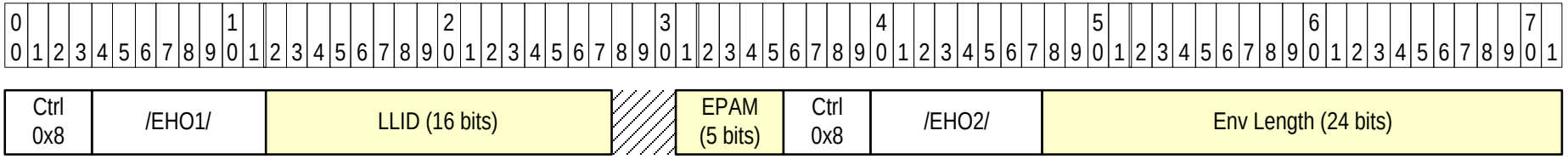
Fields:

LLID – ULID or PLID value – determines destination vMAC instance

EPAM – Envelope Position Alignment Marker

EnvLength – Envelope length in EQs

Envelope Header EQ



Thank You