

# Inner Codeword Self-sync Proposal

Xiang He, Hao Ren  
Huawei Technologies

# Supporters

Zvi Rechtman, NVIDIA

Adee Ran, Cisco

# Introduction

- Self-sync (search and test) method has been widely used in FEC codeword sync.
  - This method has been used in IEEE 802.3 Clause 74 for Firecode.
  - Self-sync was not used for RS FEC mainly because there are multiple lanes, so alignment marker is a more suitable mechanism to perform lock and deskew.
- Binary(128,120) code has been adopted for 200G/lane optical PMDs, and self-sync is suitable to find the inner FEC codeword boundaries.
  - Multi-lane alignment is not needed, because each 200G/lane PMD has its own inner FEC.
  - The codeword length is short, so there are fewer bit/symbol locations to try.
- This presentation proposes a complete self-sync method for FEC\_I sublayer, and it is independent to other TBD items.
  - Performance is evaluated with target BER range.

# Revisit: Self-sync Used for Firecode

## 74.7.4.5.1 FEC (2112,2080) decoding

The FEC decoding function block diagram is shown in Figure 74–9. The decoder processes the 16-bit rx\_data-group stream received from the PMA sublayer and descrambles the data using the PN-2112 pseudo-noise sequence as described in 74.7.4.4.1.

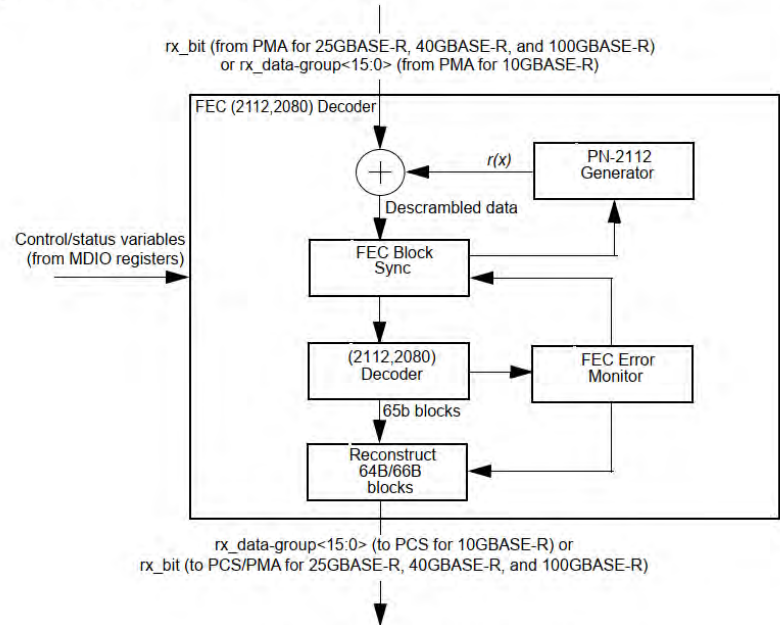


Figure 74–9—FEC (2112,2080) decoding

It can be described in three steps:

**Search and Test** — Test a position to see if it is the boundary, if not shift 1 bit.

**Validate** — Validate a tested position if this is the right boundary.

**Monitor and Drop** — See if the established sync breaks, possibly due to Hi-BER.

## 74.7.4.7 FEC block synchronization

The receive synchronization of FEC blocks is illustrated by FEC Lock state diagram in Figure 74–11.

Receive FEC block synchronization is achieved using conventional n/m serial locking techniques as described as follows:

- a) Test a potential candidate block start position
  - 1) Descramble block using PN-2112 Generator per 74.7.4.4.1
  - 2) Evaluate parity for the potential block
    - i) If the parity does not match (i.e., the received parity does not match the computed parity), shift candidate start by one bit position and try again.
- b) Validate potential block start position has good parity for “n” consecutive blocks
  - 1) If any of them fail shift candidate start one bit position and start again
  - 2) If “n” consecutive blocks are received with good parity, report Block Sync
- c) Block Sync is established.
- d) If “m” consecutive blocks are received with bad parity, drop Block Sync and restart again at item a).

The procedure is repeated at most 2111 times for all bits positions in the 2112 codeword. The values for m and n are as follows: m = 8 and n = 4.

# Revisit: Self-sync was Proposed for RS FEC in 25 GbE

- [https://www.ieee802.org/3/by/public/Mar15/slavick\\_3by\\_01a\\_0315.pdf](https://www.ieee802.org/3/by/public/Mar15/slavick_3by_01a_0315.pdf)

802.3by March 2015 Plenary

Your Imagination, Our Innovation

## Alternate method: Scramble and Test logic

- **Scramble RS-codeword after RS-encoding**
- **Descramble RS-codeword**
- **Use RS-FEC correction logic**
  - Find a CW with 0 errors, then check if next CW is correctable
  - This logic is required for normal operation
- **This is similar to the method used in Clause 74**
  - But allows for a higher CER (codeword error rate)

AVAGO  
TECHNOLOGIES

802.3by March 2015 Plenary

Your Imagination, Our Innovation

## Logic dedicated to codeword alignment

- **CWM (using parallel detect)**
  - 10k gates for parallel test and detect
  - 3k gates for Tx buffer<sup>1</sup>
  - 800 gates for Tx insertion and timer
  - 3k gates for Rx elastic buffer<sup>1</sup>
  - 150 gates for deletion and timer
- **Search and test (using single location testing)**
  - 160 gates scrambler
  - 160 gates for de-scrambler
  - Note if CI74 is required these are effectively free

<sup>1</sup> Designs which do data flow push-back wouldn't need this logic

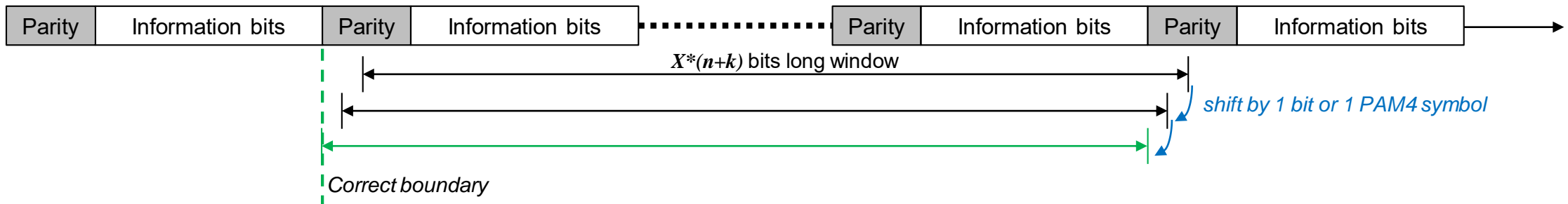
AVAGO  
TECHNOLOGIES

- Efficiency and gate cost were compared against AM locking mechanism.
- If it is achievable for RS-FEC, it will be even easier for shorter codes like the (128,120) inner FEC.
  - Number of positions needs to be tested is only ~128 compared to ~5k for KR4 FEC.
  - Considering PAM4 encoding, the test positions can be as few as **64**.

# Inner Codeword Self-Sync Proposal

\*

- **Steps:**
  - **Search and Test:** Check  $N$  codewords, see if at least  $n$  of them are correct.
    - ~~Or more preferably, we can try all 64 positions, check  $N$  codewords for each position, and pick the position that has the most correct codewords (requires less iterations).~~
    - Since decoders are already implemented, codeword test logic is essentially free.
    - Scrambler is not needed for the binary(128,120) code adopted.
  - **Validate:** See if at least  $p$  in the following  $P$  codewords are also good.
    - If so, sync established. If not, go back to search and test.
  - **Monitor and Drop:** When there are  $m$  codewords with errors in the following  $M$  codewords.
    - Same methodology as hi\_ser.



# Feasibility and Reliability of Self-sync

- At around 3E-3 BER, about 70% codewords will be error free.
  - At the correct codeword boundary, a codeword has a probability of 70% to be correct.
  - At the wrong boundary, the chance of a codeword is valid (undetected) is only  $2^{-8}$ , which is ~0.4%.
  - The dramatic difference can be used to perform a shift-and-test approach.
- The probability of locking to a wrong position can be calculated as:

$$\sum_{i=p}^P \binom{P}{i} * (2^{-8})^i * (1 - 2^{-8})^{(P - i)}$$

- The probability of losing sync during normal operation can be calculated as:

$$\sum_{i=0}^{p-1} \binom{P}{i} * (0.7)^i * (1 - 0.7)^{(P - i)}$$

# Performance of Self-sync vs AM lock(Clause 119 method)

- Assuming BER = 3E-3.
  - At higher BER level, AM lock performance will be degraded.
- **Search and Test:** Check **30** codewords for each position, pick the position with most correct codewords.
- **Validate:** See if there are **110** codewords in the following **200** codewords are also good. If so, sync established.
- **Monitor:** See if there are at least **70** codewords are not correct in the following **200** codewords. If so, drop sync.

The following table shows the performance comparison based on numbers above.

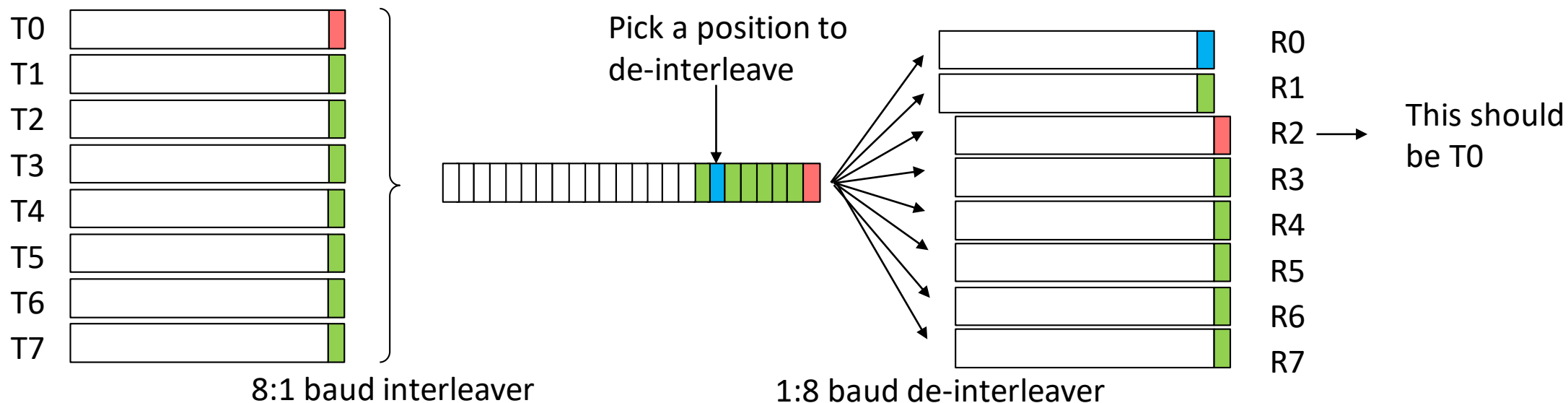
|                                    | CL119 AM           | (128,120)<br>Self-sync | Notes   |
|------------------------------------|--------------------|------------------------|---|
| Mean time to lock, $\mu\text{s}$   | 100                | <1                     | Mean time to find the codeword boundary on a bit stream. Lower is better.                             |
| Mean time to false-lock, yrs       | $6 \times 10^{22}$ | $3 \times 10^{23}$     | Mean time that it locks to a wrong position. Higher is better. <b>Should never happen.</b>            |
| Mean time to false-unlock, yrs     | $6 \times 10^{16}$ | $4 \times 10^{18}$     | Mean time that the lock breaks during normal operation. Higher is better. <b>Should never happen.</b> |
| Mean time to unlock, $\mu\text{s}$ | 400                | <1                     | Mean time to drop sync when needs to. Lower is better.  |

**Self-sync performs better in every aspect.**



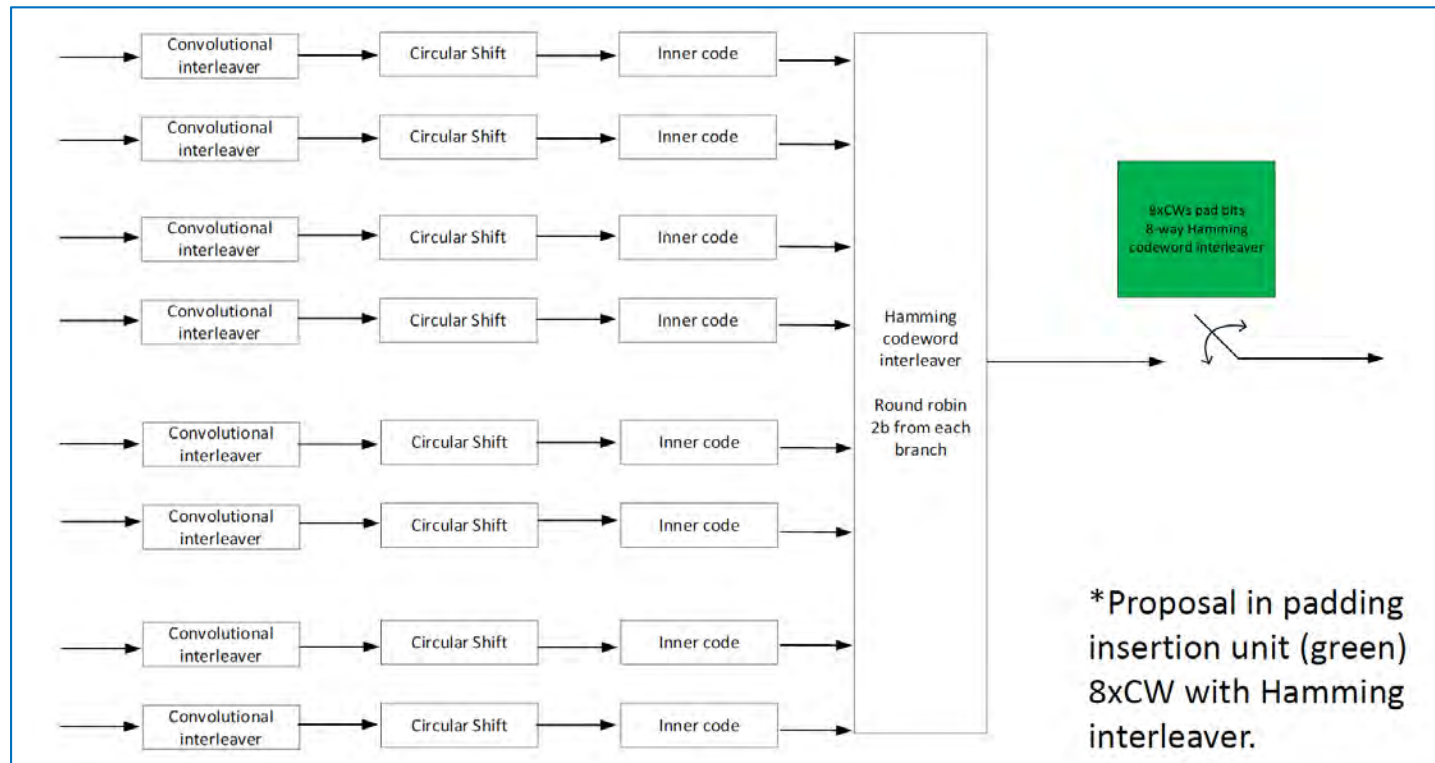
# Self-sync with Hamming Interleaver (aka Baud Interleaver)

- 8:1 baud-interleaver has been adopted for better PMD burst tolerance.
- The following steps can be used to obtain lock on the interleaved data stream.
  - Blindly de-interleave the 200G/lane data stream and obtain the 8 codeword flows.
  - Perform self-sync on de-interleaved codeword flows.
  - Find out where there is a 1 PAM4 symbol skew between two adjacent flows – the codeword boundary on the faster flow will be the correct de-interleaving (relative) position.
    - If there is no skew between the codewords flows, the de-interleaving position is already the correct position.



# Self-sync with Padding Codewords Insertion in 8x128b

- [rechtman\\_3dj\\_01\\_2305](#) proposed to insert pad block in groups of 8 CWs and baud-interleaved for better protection.
- Specially defined pattern can be then used to identify the padding, see [huang\\_3dj\\_01\\_2307](#).
  - Fixed pattern compare at known position is easier than searching for the pattern in the data stream.

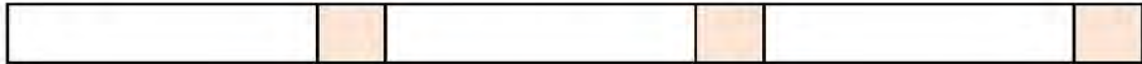


# Self-sync with Padding Codewords Insertion in 3x128b

- Original padding proposal was to insert pad block in groups of 3 CWs without interleaving.

## Padding Specification

- **384 bits = 3 CW using 128, 120 code**
  - Payload bits = 360 (=45 B), parity = 24 bits



See [farhood\\_3dj\\_01a\\_230206](#)

- Self-sync can be performed on the padding CWs directly.
  - We can perform self-sync on the 200G/lane data stream, searching for valid codewords.
  - Padding CWs are stand-alone (128,120) CWs.
  - All payload CWs are 8:1 interleaved, and can be seen as invalid codewords.

# Summary

- A self-sync method is proposed for inner FEC.
- FEC codeword boundary detection is inherent, no additional logic required.
- It performs better than AM lock mechanism.
- The self-sync method itself is “protocol agnostic”
  - It does not rely on padding content definition
  - It could be used for the two different padding insertion proposals.

Thank you