802.3dj D2.3 Comment Resolution Logic Topics

Gary Nicholl (Cisco), Logic Track Lead Editor
Matt Brown (Qualcomm), 802.3dj Chief Editor
Eugene Opsasnick (Broadcom), Logic Editor

Introduction

- This slide package was assembled by the 802.3dj editorial team to provide background and detailed resolutions to aid in comment resolution.
- Specifically, these slides are for the various logic-topic comments.

Hi_SER Calculation

Comments #3

Hi_SER Calculation - 1
Comment #3

Cl **175** *SC* **175.2.5.3** *P* **298** *L* **32** # 3

Maniloff, Eric Ciena

Comment Type **TR** *Comment Status* **D** *hi_ser (L)*

Currently hi_ser is defined as being calculated based on the number of symbol errors detected in consecutive non-overlapping blocks of 8192 codewords for 1.6TBASE-R.

In 400GbE, hi_ser was based on 8192 codewords. In 800GbE, hi_ser was based on 8192 codewords per 400G flow with the output of the two flows OR'd to report hi_ser. 100GE and 200GE hi_ser is calculated over 2 AM periods. For 400GbE & 800GbE the interval for measurement is equal to an AM period, hence no counter is required. 100GbE and 200GbE have hi_ser intervals = 2 times the AM period.

Currently 1.6TbE has changed the interval to be less than the AM period. The hi_ser measurement window should be aligned with an AM period, this will maintain the 104.8576 µs time interval used in previous the PCS of 400 & 800GbE.

SuggestedRemedy

Two resolutions are possible:

Option 1: Change the hi_ser measurement interval to 32768 codewords (4x8192) to be consistent with the AM period, with the threshold for declaring hi_ser scaled by a factor of 4.

Option 2: Follow the approach of 800GbE, and calculate hi_ser individually in each FEC decoder over 8192 codewords with the results OR'd.

Proposed Response *Response Status* **W**

PROPOSED REJECT.

This comment does not apply to the substantive changes between IEEE P802.3dj D2.2 and D2.3 or the unsatisfied negative comments from previous drafts. Hence it is not within the scope of the recirculation ballot.

In addition, the comment makes a connection between the alignment marker insertion period and the hi_ser computation period that does not actually exist. There is no logical connection between the AM period and the hi_ser calculation.

The hi_ser calculation has been consistently performed over a block of 8192 codewords from 50GbE through 1.6TbE port speeds. The important metric for asserting the hi_ser indication is the percentage of RS-FEC symbols in error, and both options given in the suggested remedy do not change this value.

The comment does not provide sufficient justification to support the suggested remedy.

Hi_SER Calculation - 2
Comment #3

- In Clause 175, hi_ser is asserted when 5560 symbol errors occur within a window of 8192 RS-FEC codewords (across all 4 FEC decoders)

- The comment is suggesting two possible changes:
  Option 1: Change to 22,240 symbol errors in a window of 32,768 codewords across all 4 FEC decoders
  Option 2: Change to 5560 symbol errors in a window of 8192 codewords within each of the 4 FEC decoders and then logical-OR the four separate indications to create a single hi_ser indication.

Hi_SER Calculation - 3
Comment #3

Port Speed	Hi_ser window (codewords)	Hi_ser threshold (symbols)	Hi_ser window (ns)	AM interval (CWs)	AM interval (ns)
50GbE	8192	6380	838860.8	1k	104857.6
100GbE (1x100)	8192	5560	419430.4	4k	209715.2
200GbE	8192	5560	209715.2	4k	104857.6
400GbE	8192	5560	104857.6	8k	104857.6
800GbE (per flow)	8192	5560	104857.6	8k	104857.6
1.6TbE	8192	5560	26214.4	32k	104857.6

Hi_SER Calculation - 4
Comment #3

Editorial suggestion: No change

- The draft is technically correct as written.
- The hi_ser window (8192 codewords) and threshold (5560 symbols) are consistent with 100/200/400/800GbE standards.
- The suggested remedy has no technical benefit.
  - The suggested remedy makes no change to the percentage of symbol errors to set hi_ser.
  - A wider window delays setting hi_ser and increases probability of false packet acceptance.
- The potential implementation savings of the suggested remedy are insignificant and do not justify a change at this time in the process.
  - Implementations must have a counter to determine the expected AM spacing. This same counter can be used to indicate quarter AM periods.

Inner FEC Error Bin Counters

Comment #150

Inner FEC Error Bin Counters Comment #150

Cl **177** *SC* **177.5.5** *P* **366** *L* **43** # 150

Slavick, Jeff — Broadcom

Comment Type **TR** *Comment Status* **D** *bin counters (L)*

The definition of the error_bin calls for the bin to contain CW with k bits "corrected". I believe the intent is that error bin 0 is to count CW received with 0 errors in them.
However, an uncorrected CW could also have 0 corrected bits as the decoder couldn't figure out what to do and thus changed nothing.
I believe we want corrected_cw + uncorrected_cw + error_bin_0 = cw_counter. And that error_bin_1 + error_bin_2 + error_bin_3 = corrected_cw.
So we should explicity call out that error_bin_0 is a CW received with 0 errors.

SuggestedRemedy

In 177.5.5 and 184.5.7 add the following to the definition of Inner_FEC_codeword_error_bin_k:

Error bin 0 increments when the Inner FEC codeword contains no detected bits in error.

Proposed Response *Response Status* **W**

PROPOSED ACCEPT IN PRINCIPLE.

The current wording is ambiguous for how to count codewords in bin_0. It could be interpreted to mean that codewords that are detectd as "uncorrectable" as well as "correct" should be counted since no bits are flipped. The intention for adding bin_0 was to count codewords determined to be correct. Codewords that are determined to be uncorrectable are already counted in Inner_FEC_uncorrected_cw_counter and should not be double counted. The changes needed to make this clear should not change the intended behavior, but only clear up this ambiguity..

In 177.5.5, in the definition of Inner_FEC_codeword_error_bin_k,
Change:
"While Inner_FEC_sync_status is true, Inner_FEC_codeword_error_bin_k counts once for each codeword received with exactly k bit corrected (flipped). For example, if an Inner FEC codeword has exactly two bits corrected, the Inner_FEC_codeword_error_bin_2 is incremented."
To:
"While Inner_FEC_sync_status is true, Inner_FEC_codeword_error_bin_k counts once for each codeword received with exactly k detected incorrect bits. For example, if an Inner FEC codeword has exactly two bits detected in error, the Inner_FEC_codeword_error_bin_2 is incremented. Note that Inner_FEC_codeword_error_bin_0 is only incremented for codewords determined to be correct by the decoder (zero incorrect detected bits) and is not incremented for codewords determined to be uncorrectable which are counted in Inner_FEC_uncorrected_cw_counter."

In 184.5.7, in the definition of Inner_FEC_codeword_error_bin_k,
Change:
"While alignment_valid is true, for each Inner FEC codeword received with exactly k corrected (flipped) bits, Inner_FEC_codeword_error_bin_k is incremented. For example, if an Inner FEC codeword has exactly two bits corrected, then

Inner_FEC_codeword_error_bin_2 is incremented."
To:
"While alignment_valid is true, Inner_FEC_codeword_error_bin_k counts once for each codeword received with exactly k detected incorrect bits. For example, if an Inner FEC codeword has exactly two bits detected in error, the Inner_FEC_codeword_error_bin_2 is incremented. Note that Inner_FEC_codeword_error_bin_0 is only incremented for codewords determined to be correct by the decoder (zero incorrect detected bits) and is not incremented for codewords determined to be uncorrectable which are counted in Inner_FEC_uncorrected_cw_counter."

Inner FEC Error Bin Counters
Comment #150

- Comment #150 points out that the description of error bin_0 is ambiguous. It could be interpreted to mean that codewords that are detected as "uncorrectable" as well as those with “no errors” should be counted since no bits are “corrected” in both of these cases. Uncorrectable codewords have a separate counter and should not be double counted.
  - This is addressed by changing the terms “corrected bits” to “detected bit errors”
- In addition, the definition is self-contradictory for the last bin. It states that bin_k is incremented when exactly k bits are corrected (in error). But also states that the last bin is incremented when “k or more bits” are corrected.
- The original proposed change fixed the first issue. As the response was refined this week, the second issue came up and is now also addressed.
- The new updated response also affects comment #149 which is in the bucket.
  - The new response for #150 must superseed the response to comment #149.

Inner FEC Error Bin Counters - Current definition in 177.5.5 Comment #150

Inner_FEC_codeword_error_bin_*k*

A set of four 32-bit counters where k = 0 to 3. While Inner_FEC_sync_status is true, Inner_FEC_codeword_error_bin_k counts once for each codeword received with exactly k bits corrected (flipped). For example, if an Inner FEC codeword has exactly two bits corrected, then Inner_FEC_codeword_error_bin_2 is incremented. Error bin 3 increments when three or more bits are corrected in an Inner FEC codeword.

Issue 1: CWs with no errors and uncorrectable CWs both correct zero bits. Fix: Change “k bits corrected” to “k bits detected in error”. Also add additional explanation for bin_0.

Issue 2: error_bin_3 is defined for both “exactly 3 bits corrected” and when “3 or more bits” are corrected. Fix: Separate the definitions for k=0 to 2, and for k=3.

Inner FEC Error Bin Counters - Current definition in 184.5.7 Comment #150

Inner_FEC_codeword_error_bin_*k*

A set of five 32-bit counters where *k* = 0 to 4. While alignment_valid is true, for each Inner FEC codeword received with exactly *k* corrected (flipped) bits, Inner_FEC_codeword_error_bin_*k* is incremented. For example, if an Inner FEC codeword has exactly two bits corrected, then Inner_FEC_codeword_error_bin_2 is incremented. Note that bin 4 is for 4 or more bits corrected in an Inner FEC codeword.

Issue 1: CWs with no errors and uncorrectable CWs both correct zero bits. Fix: Change “k bits corrected” to “k bits detected in error”. Also add additional explanation for bin_0.

Issue 2: error_bin_4 is defined for both “exactly 4 bits corrected” and when “4 or more bits” are corrected. Fix: Separate the definitions for k=0 to 3, and for k=4.

Inner FEC Error Bin Counters
Comment #150

New wording for definition of Inner_FEC_codeword_error_bin_k in subclause 175.5.5:

A set of four 32-bit counters, where $k = 0$ to 3. When Inner_FEC_sync_status is true, for $k = 0$ to 2, Inner_FEC_codeword_error_bin_k counts once for each codeword received with exactly k bits detected in error. Inner_FEC_codeword_error_bin_3 increments when Inner_FEC_sync_status is true and three or more bits in an Inner FEC codeword are detected in error and are corrected. For example, if an Inner FEC codeword has exactly two bits detected in error, then Inner_FEC_codeword_error_bin_2 is incremented. Note that Inner_FEC_codeword_error_bin_0 is only incremented for codewords determined to be correct by the decoder (zero detected incorrect bits) and is not incremented for codewords determined to be uncorrectable which are counted in Inner_FEC_uncorrected_cw_counter.

Inner FEC Error Bin Counters
Comment #150

New wording for definition of Inner_FEC_codeword_error_bin_k in subclause 184.5.7:

A set of five 32-bit counters, where *k* = 0 to 4. When alignment_valid is true, for *k* = 0 to 3, Inner_FEC_codeword_error_bin_*k* counts once for each codeword received with exactly *k* bits detected in error. Inner_FEC_codeword_error_bin_4 increments when alignment_valid is true and four or more bits in an Inner FEC codeword are detected in error and are corrected. For example, if an Inner FEC codeword has exactly two bits detected in error, then Inner_FEC_codeword_error_bin_2 is incremented. Note that Inner_FEC_codeword_error_bin_0 is only incremented for codewords determined to be correct by the decoder (zero detected incorrect bits) and is not incremented for codewords determined to be uncorrectable which are counted in Inner_FEC_uncorrected_cw_counter.

Thank you