Thoughts on the 800 Gb/s and 1.6 Tb/s PCS

P802.3df Task Force, 03 February 2022

Tom Huber (Nokia)

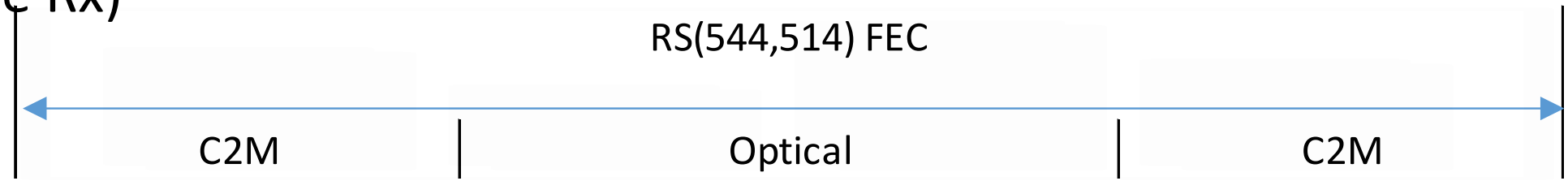
Steve Trowbridge (Nokia)

FEC for 800G and 1.6T interfaces

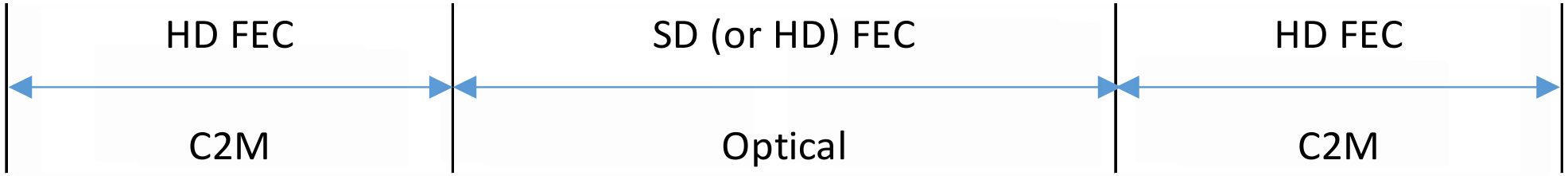
- While 802.3bs initially specified a single RS(544,514) FEC code, this has evolved to include additional FEC codes in subsequent projects
- It is almost certain that multiple FEC codes will be selected for 800 Gb/s and 1.6 Tb/s operation by P802.3df and follow-on projects at these rates
  - While RS(544,514) may be adequate for early-adopter PHYs with 100 Gb/s lanes, higher coding-gain FEC may be required for PHYs with 200 Gb/s lanes. Several possible higher-gain FECs were mentioned in SG presentations, e.g., higher-order RS and HD or SD BCH codes described in he_b400g_01_210426.pdf
  - Some PHYs will likely be based on coherent interfaces, which generally use different families of FEC codes with higher NCG. 802.3ct uses Staircase FEC, P802.3cw uses a concatenated CFEC and SD FEC, OIF has selected OFEC for 800 Gb/s coherent interfaces
- Different FEC codes have significantly different codeword sizes
  - RS(544,514) has a 5,140-bit codeword
  - Staircase FEC has a 261,120-bit codeword

FEC architectures used currently in 802.3

- 802.3bs defines a single end-to-end RS(544,514) FEC code (i.e., from the PCS on the host board of the Tx to the PCS on the host board of the Rx)

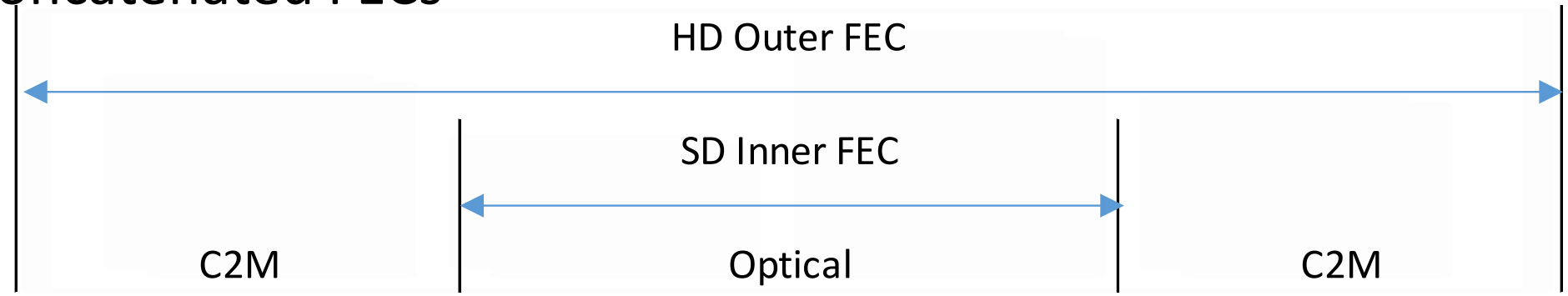


- 802.3ct and P802.3cw use a cascaded FEC architecture, where the FEC code that is added on the host board is terminated at the optical module and a different FEC code is added by the optical module

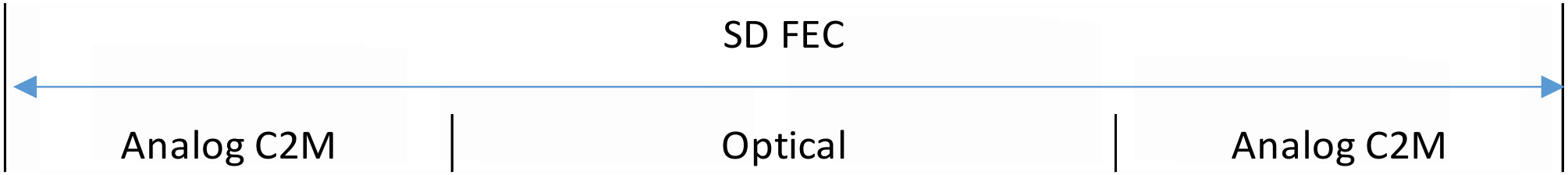


Other FEC architectures that may need to be considered for B400G PHYs

- Concatenated FECs

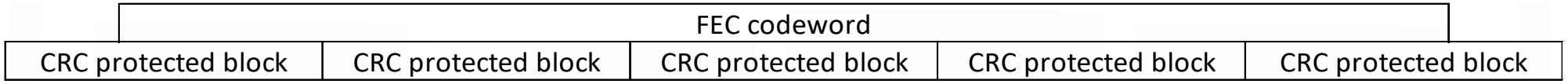


- If ACO modules are used, a single soft decision FEC code can be used end-to-end (see for example OIF-COM-ACO-1.0.pdf).
  - Idea is that the C2M interface is an Analog IQ interface for the two polarizations, and that the Tx DAC and Rx ADC are on the host board. The same coherent modulation and SD FEC can be used end-to-end.



Error marking

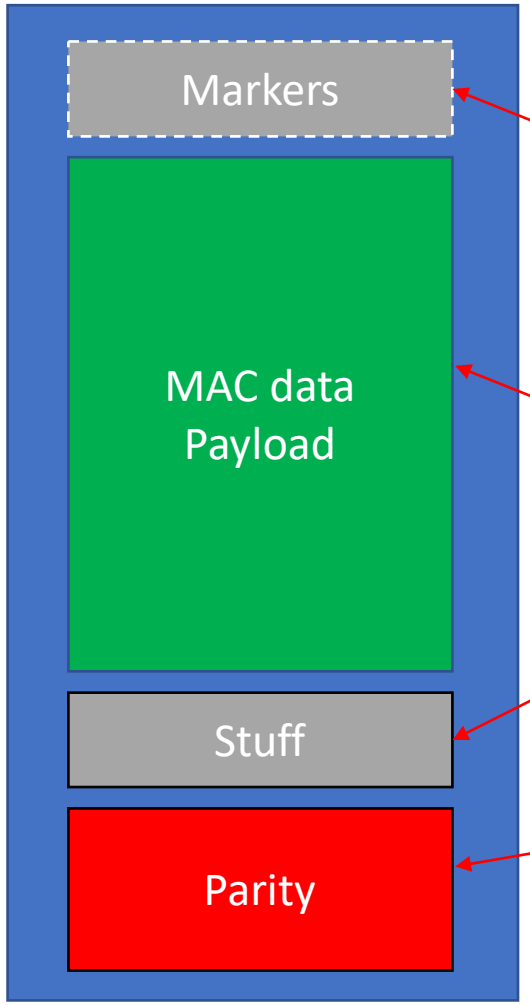
- Some hard-decision FEC codes have high confidence of knowing that a FEC codeword is uncorrectable, but most soft decision FEC codes do not and would require an ‘inner CRC’ to detect uncorrected errors
  - The ‘inner CRC’ would cover some amount of PCS data
  - The purpose of the ‘inner CRC’ is to detect when the FEC was unable to correct errors so that the PCS blocks can be error-marked
    - As a bonus, this mechanism would also enable error-marking when the PCS is carried over an OTN link, in support of the objective to provide support for mapping over OTN
- If a common ‘inner CRC’ is used across FECs, because FEC codewords have different sizes, the unit of PCS data that is protected by the ‘inner CRC’ may not align with the FEC codeword boundary



Mapping PCS data to FEC codewords

- RS(544,514) FEC has a fixed relationship of 257B blocks of PCS data to FEC codewords
  - 66B blocks are transcoded to 257B, and twenty 257B blocks form a FEC codeword
- Other FECs allow the Ethernet payload to ‘float’ in the FEC frame and use a ‘stuffing mechanism’ for rate compensation
  - Staircase FEC in 802.3ct, and CFEC in P802.3cw, both use GMP for this purpose

Information content of a FEC codeword (conceptual view)



A subset of the codewords may contain markers that allow the Rx to find the start of FEC codewords and to deskew across lanes that form the FEC codeword

MAC frame data is encoded in this area. The coding must allow locating the start/end of MAC frames and signaling LF/RF when the link is down, indicating /E/ control characters, and /I/ or /LPI/ between packets

A variable amount of stuff bits or bytes may exist in the frame that can compensate MAC/PHY rate differences

The redundancy that the FEC adds is in this area.

- IEEE Std 802.3ct and draft P802.3cw both use GMP to map 66B PCS data into a FEC frame, with the stuff distributed evenly throughout the MAC data Payload (rather than a separate field)
- Other stuff mechanisms (e.g., the “AMP” style of mapping with fixed positions that could be data or stuff) can be considered based on codeword size and maximum MAC/PHY clock rate differences
- The RS(544,514) FEC frame can also be described using this structure – the ‘stuff’ area always contains zero bytes because the MAC data Payload area is defined to be twenty 257B codewords

Summary of FEC considerations

- It should be assumed that multiple FEC codes will be needed
- The ability to support multiple FEC architectures should not be precluded
- In order to provide a single mechanism for error marking across all 800G and 1.6T PHYs, an ‘inner CRC’ should be used
  - Since FEC codeword size varies widely between FEC codes, it should be assumed that the inner CRC block is not locked to the FEC codeword
  - More analysis is needed to determine how much PCS data should be included in an inner CRC block
- Due to significant differences in FEC codeword sizes, it should be assumed that it is not possible to maintain a fixed relationship between PCS data and FEC codewords
  - It should be assumed that the Ethernet payload will ‘float’ within the FEC codeword

Considerations for the 800G and 1.6T PCS

- Lane striping
- Rate adaptation
- PCS coding
- Economy of transmission
- Start of packet alignment
- Clock tolerance

Lane striping

- The optimal lane striping depends on the modulation (e.g., PAM4, PAM6, DP-16QAM), the physical lane count, and the FEC code.
- As striping into physical lanes likely occurs after the FEC encoder, and there will likely be multiple FEC codes, there may be multiple lane striping methods used for different P802.3df PHYs
- It is proposed that the P802.3df Task Force assume that multiple lane striping methods will be used

Rate adaptation

- Historically, rate adaptation has been accomplished via inserting and deleting Idle characters from the data stream in groups that match the start of packet granularity
- There are multiple issues with this approach, particularly for high-speed links:
  - Idles are irregularly spaced
  - Requires looking at the data stream to find where the Idles are
  - Inserting or removing Idles creates problems with PTP accuracy
- It is proposed to use the ‘float’ of the Ethernet payload within the FEC frame to provide rate compensation at high speed, and not do Idle insertion/deletion
  - Enables definition of a common way to associate PCS data to FEC codewords for all FEC codes
  - Allows the interPacketGap parameter (IEEE 802.3 table 4.2) to be set at zero for MAC data rates of 800 Gb/s and 1.6 Tb/s, as Idles between packets are not needed for rate compensation

PCS coding

- The 66B code was originally used to provide resilience in order to meet MTTFPA requirements on links that have Poisson-distributed errors
- None of the links used for B400G PHYs (or 200G and 400G PHYs, or most 100G PHYs) have Poisson-distributed errors due to the use of FEC and decision-feedback equalizers
  - The transcoding to 257B used prior to adding FEC discards the elements of the 66B code that provide resilience since they are not needed
  - OTN links also all have FEC and DFE, so the resilience of the 66B code is neither needed nor useful on those links
- It should not be assumed that 800G and 1.6T PHYs must use the 66B code
  - The resilience of the 66B code is not needed
  - Describing the PCS in terms of 66B coding, followed immediately by transcoding to 257B (or perhaps something more efficient) adds complexity for the reader
  - Wider start of packet granularity may be desirable, making 66B coding problematic

Economy of transmission

- It has often been observed that there are a lot of “wasted bits” carried across Ethernet interfaces, including the idle/IPG and the preamble, which is primarily useful for CSMA/CD collision detection on half-duplex interfaces
- While it is likely necessary to preserve the Start character and the Preamble at the service interface for the Reconciliation Sublayer (RS) to provide a common view toward the MAC and bridging layers, it wouldn’t be necessary to transmit these bytes across the physical interface itself
  - Note that frame-transparent OTN mappings already do this (the GFP header replaces the preamble and Start character and provides frame delineation)
- It is proposed that the P802.3df Task Force consider dropping the preamble and start characters from the set of bits/bytes transported across the PHY
  - This does not change the MII or anything above it
  - While this would reduce the bit rate slightly, it would also introduce some complexity since the number of bits discarded is fixed, but the size of a packet is not

Start of packet granularity

- The Clause 49 66B code with 4-byte Start of Packet alignment was introduced in 2002 by the P802.3ae project for MAC data rates of 10 Gb/s (and then re-used for 2.5G, 5G, 25G)
- The Clause 82 66B code with 8-byte Start of Packet alignment was introduced in 2010 by the P802.3ba project for MAC data rates of 40 Gb/s and 100 Gb/s (reused for 50G, 200G, 400G)
- Higher speed designs are generally the result of combining higher clock rates and wider busses, both of which are enabled by progressively smaller CMOS process nodes
  - As busses get wider, choosing a coarser granularity for start of packet reduces the need for large barrel shifters in designs, but also reduces efficiency
  - There are several ways the loss in efficiency could be mitigated
    - Choosing a more efficient coding than 257B
    - Not using Idle insert/delete for rate adaptation
    - Discarding the preamble in the Tx PCS and reinserting it in the Rx PCS (so the MII is unchanged, but the bytes of the preamble are not carried over the link since they are not necessary for frame delineation)
- 800G is 20 times the speed of the lowest speed link that uses 8-byte alignment; in comparison, the transition from 4-byte alignment to 8-byte was made at 100G, 10 times faster than 10G (the only link using 4-byte alignment at that time)
- It is proposed to select a start of packet granularity greater than 8 bytes (specific value TBD) for 800 Gb/s and 1.6 Tb/s operation
  - An informal poll can be taken among developers to ascertain what typical (device internal) bus widths are likely to be used in devices at these speeds, such that an optimal value can be chosen that reduces the need for barrel shifters in designs

Clock Rate Tolerance

- Historically, Ethernet has chosen a clock-rate tolerance (MAC and PHY) of ±100ppm.
- As speeds get higher, clock and data recovery becomes easier with a narrower clock tolerance range.
- Telco interfaces have typically chosen ±20ppm. Clock generation within this tolerance range is very much commodity and not expensive.
- IEEE Std 802.3ct reuses the ITU-T G.709.2 frame format and inherits its ±20ppm clock tolerance for the PHY, while still allowing ±100ppm variation at the MAC.
- The P802.3ck project has elected to limit the clock tolerance for 100 Gb/s Electrical lanes to ±50ppm. Since P802.3df will specify at least some PHYs with 200 Gb/s electrical lanes, it is likely desirable to reduce the clock tolerance range still further.
- It is proposed that the P802.3df adopt a clock tolerance range of ±20ppm, leveraging the existing technology base for clocks operating within this tolerance.

Summary of proposals

- Assume multiple FEC codes and FEC architectures will be needed
- Use an ‘inner CRC’ to determine if there are uncorrectable FEC errors since multiple FECs will be used and it cannot be assured that all FECs can detect that there are uncorrected errors
- Assume lane striping may vary based on modulation, FEC, and physical lane count, and therefore needs to be specified per-PHY
- Perform MAC/PHY rate compensation via the mapping into the FEC frame rather than via Idle insertion/deletion
  - Reduce the interPacketGap to zero since Idles are not required to be available for rate adaptation
- Consider not transmitting the preamble and SFD
- Select a start of packet alignment granularity > 8 bytes (specific granularity TBD) and select an appropriate coding based on that alignment granularity
  - No need to retain 66B coding since all links will use FEC
- Specify clock tolerance of ±20ppm

THANKS!