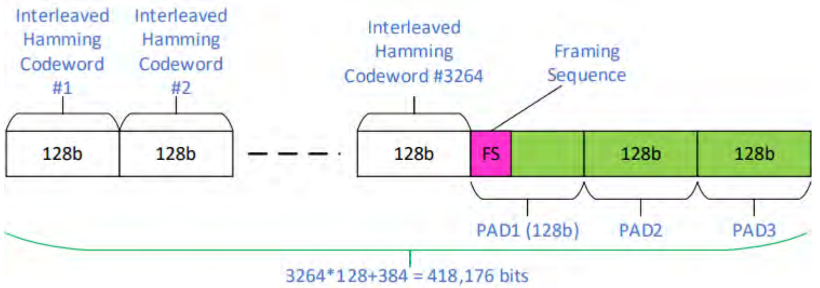
Consideration on Framing Sequence for Type 2 Inner FEC

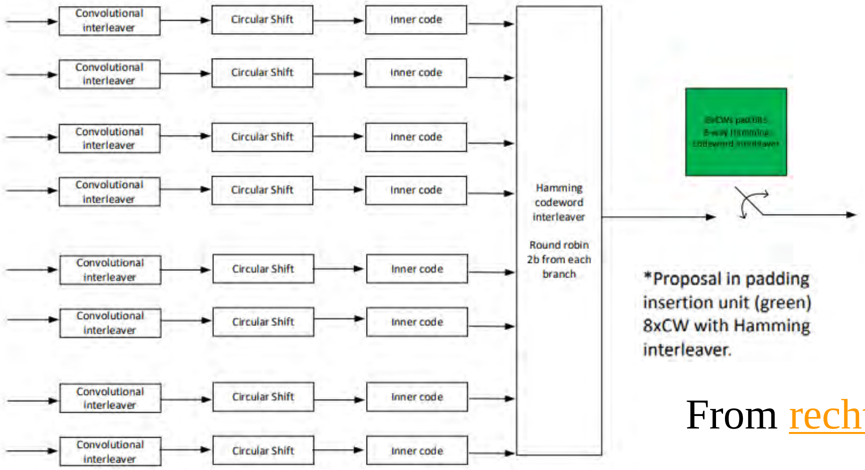
Kechao Huang, Xiaoling Yang, Qinhui Huang, and Huixiao Ma
Huawei Technologies

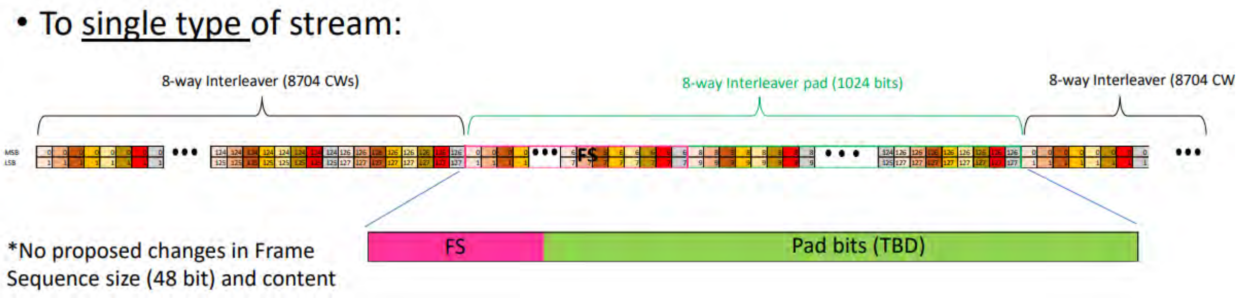
Background

- Type 2 PHY/FEC with Inner(128,120) code and padding was adopted
  - See patra_3dj_01b_2303, and motion #5 in motions_3dfdj_2303 in March 2023
  - Insert 384 padding bits after every 3264 Hamming codewords → 113.4375GBaud
  - Framing Sequence (FS) included in padding bits for synchronization in receiver
- FS synchronization was analyzed in barakatain_3dj_01a_2303
  - Consider 48-bit FS (0x9A, 0x4A, 0x26, 0x65, 0xB5, 0xD9) same as the 200G/400G/800G PCS AM (common marker portion)
  - Suggest the FS lock process: Each FS lock looks for *n*=3 valid FS, and Out of FS Lock is when *k*=6 invalid FS observed
- Simplified pad insertion was proposed in rechtman_3dj_01a_2305
  - Propose to insert 1024 (8 × 128) padding bits after every 8704 Hamming codewords, in addition to Hamming interleaver protection
  - In order to allow possible implementation of Search & Test synchronization



From patra_3dj_01b_2303





From rechtman_3dj_01a_2305

This presentation

- Provide detailed FS lock process including FS lock state diagram for future draft document
- Propose to improve the FS format for hardware reuse purpose
- Analyze the Search & Test synchronization method

FS lock process

- Recap on the analysis on FS lock process
  - Consider the 48-bit FS (0x9A, 0x4A, 0x26, 0x65, 0xB5, 0xD9) same as the common marker (CM) portion of the 200G/400G/800G PCS AM*
  - The FS divided into 12 half-byte nibbles; If *m*=9 or more nibbles in the candidate block match the corresponding known nibbles in the FS portion, the candidate block is considered a valid FS+
  - Each lock looks for *n*=3 valid FS, and Out of Lock if *k*=6 invalid FS observed, for guaranteeing true lock with very high probability, with expected time to failure > AOU

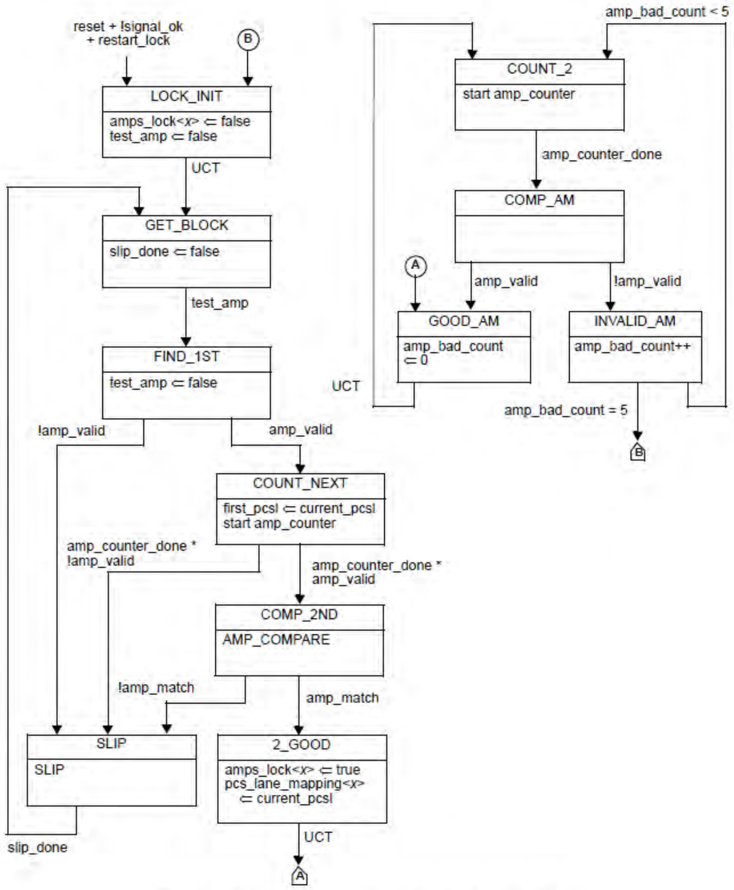


Figure 119–12—Alignment marker lock state diagram

From IEEE 802.3 Clause 119

- Probability Calculation:
  - Probability of falsely locked P_{fl}: equals to p_{fl}^{n}, where $p_{fl} = \sum_{i=m}^{12}\binom{12}{i}(1-p_0)^i * p_0^{12-i}$, with $p_0 = 15/16$ corresponding to a mismatched nibble.
  - Probability of falsely unlocked P_{fu}: equals to p_{fu}^{k}, where $p_{fu} = \sum_{i=0}^{m-1}\binom{12}{i} * p_1^{12-i} * (1-p_1)^i$, with $p_1 = 1-(1-\text{BER})^4$, where BER=4.8e-3 is assumed in the tables below.
  - Mean time to truly locked state is roughly estimated by (n -0.5)×group delay, where group delay corresponds to ~1.8μs (418176 bits)

		Mean time to false alignment (years)		
threshold m	P_fl	n=2	n=3	n=4
12	3.55E-15	1.16E+15	3.26E+29	9.17E+43
11	6.43E-13	3.57E+10	5.59E+22	8.74E+34
10	5.34E-11	5.25E+06	9.95E+16	1.89E+27
9	2.69E-09	2.10E+03	7.96E+11	3.02E+20
8	9.17E-08	1.84E+00	2.07E+07	2.33E+14
7	2.23E-06	3.20E-03	1.50E+03	7.01E+08

		Mean time to false unlock (years)			
threshold m	P_fu	k=3	k=4	k=5	k=6
6	6.58E-10	2.05E+14	3.12E+23	4.74E+32	7.21E+41
7	4.02E-08	9.03E+08	2.25E+16	5.60E+23	1.39E+31
8	1.78E-06	1.03E+04	5.79E+09	3.25E+15	1.82E+21
9	5.78E-05	3.02E-01	5.23E+03	9.05E+07	1.57E+12
10	1.34E-03	2.43E-05	1.82E-02	1.36E+01	1.01E+04
11	2.11E-02	6.20E-09	2.94E-07	1.39E-05	6.58E-04
12	2.06E-01	6.66E-12	3.23E-11	1.57E-10	7.60E-10

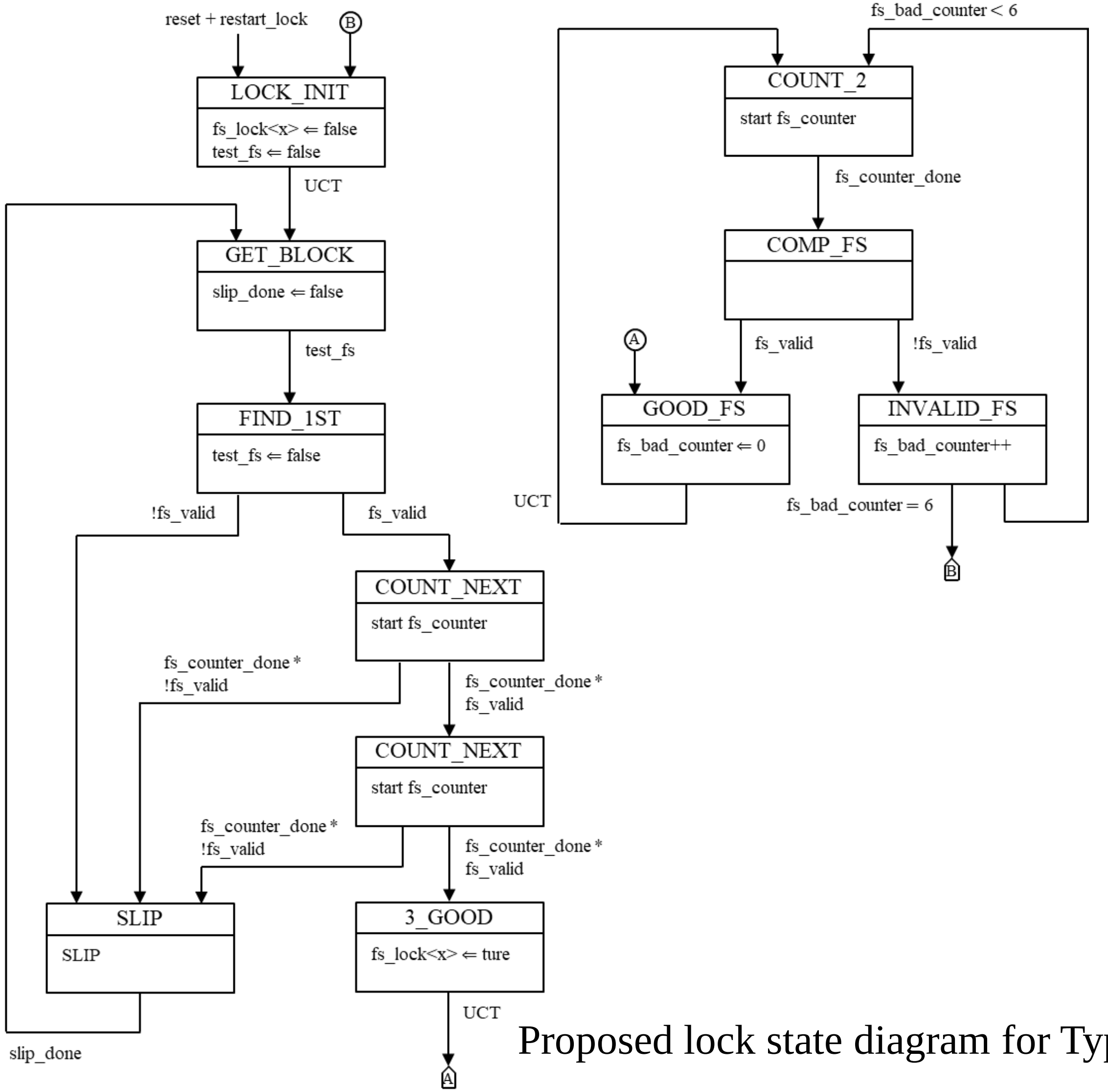
Remark: highlighted with green represent greater than AOU.

From barakatain_3dj_01a_2303

*The 0x9A4A2665B5D9 in previous contributions is typo. Note that each octet is transmitted LSB to MSB.

+This process is same as the CM sync, see the amp_valid description in "IEEE 802.3-2022 119.2.6.2.2 Variables"

FS lock process: lock state diagram

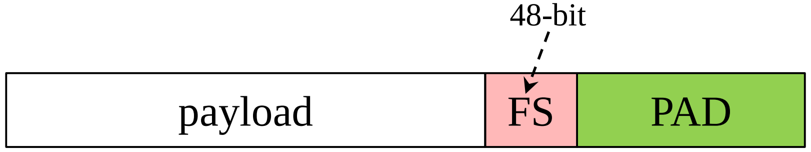


Proposed lock state diagram for Type 2 PHY/FEC

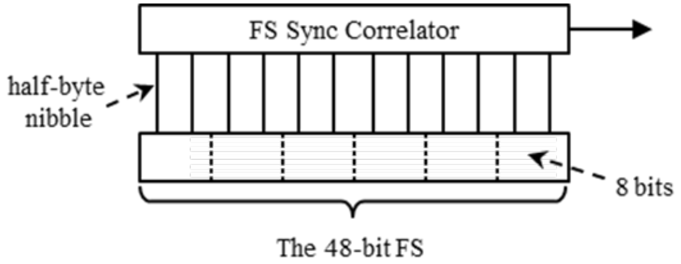
- The 800G receiver (with 4 lanes) shall implement 4 FS lock processes and an FS process operates independently on each lane
- The FS lock can identify both the Hamming codeword boundary and the location of padding bits

Proposed FS format (1/2)

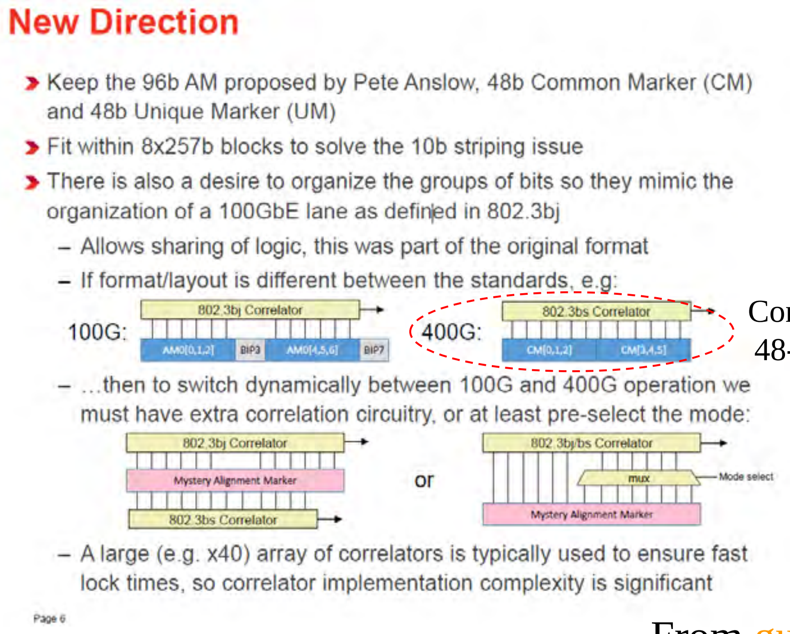
- Essential function in the FS lock process:
  - The 48-bit FS portion are compared on a nibble-wise basis (12 comparisons)
  - The FS are continuous in the bits stream, which may not be hardware reuse friendly
- Propose to organize the groups of FS
  - Mimic the organization of the 200G/400G/800G PCS AM, and allow the logic reuse
  - The 48-bit FS consist of two groups, each with three bytes
  - There is a one byte gap between the two groups.



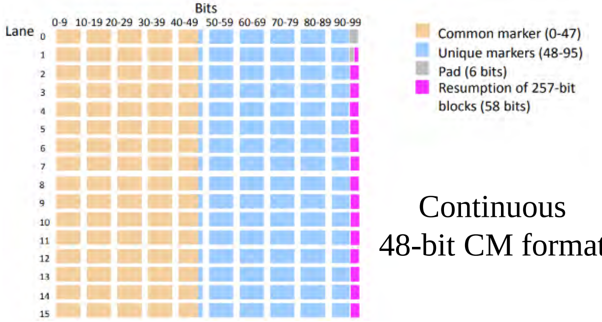
FS format in patra_3dj_01b_2303



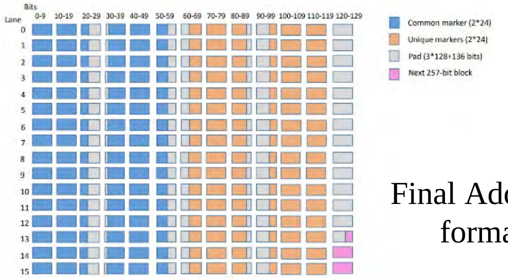
Corresponding FS sync correlator



Continuous 48-bit CM

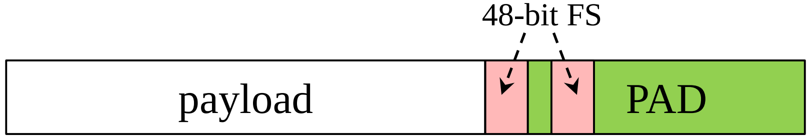


Continuous 48-bit CM format

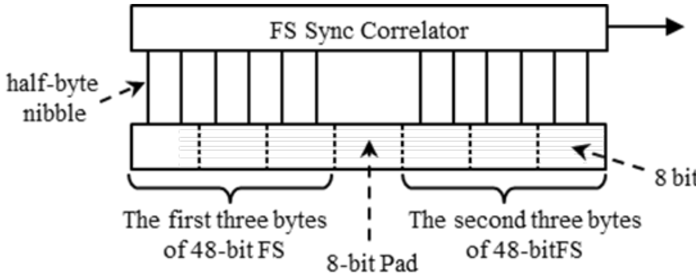


Final Adopted format

From gustlin_01_0216_logic



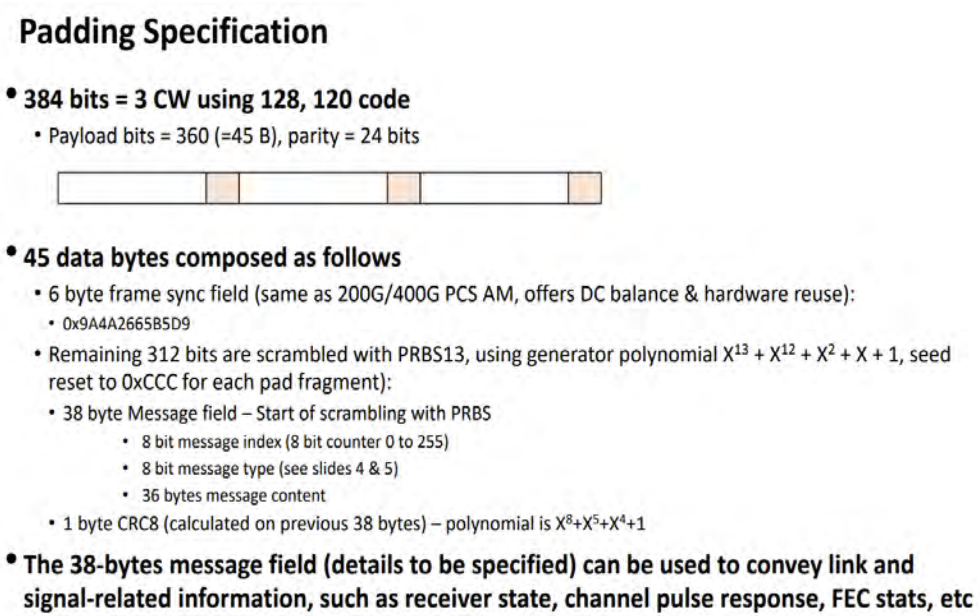
Proposed FS format



Corresponding FS sync correlator

Proposed FS format (2/2)

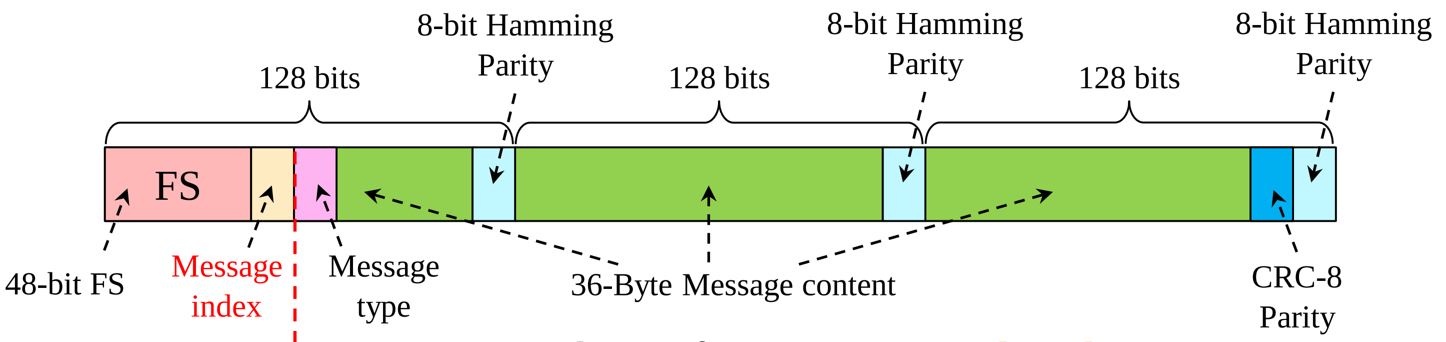
- The effect on the padding specification*
  - The position of “1-byte Message index” was changed to “be located between two 3-byte FS groups”
  - The size of “1-byte Message index”, “1-byte Message type”, and “36-byte Message content” remain the same



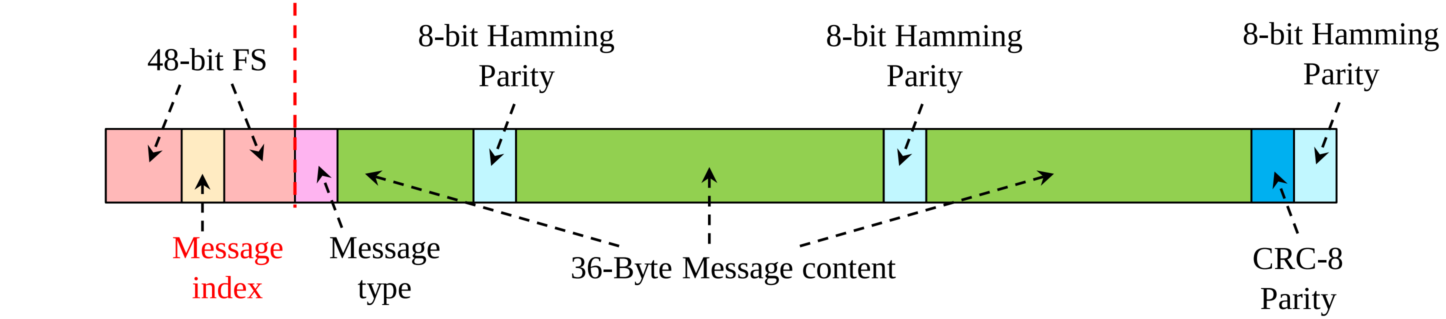
Padding Specification

- **384 bits = 3 CW using 128, 120 code**
  - Payload bits = 360 (=45 B), parity = 24 bits
- **45 data bytes composed as follows**
  - 6 byte frame sync field (same as 200G/400G PCS AM, offers DC balance & hardware reuse):
    - 0x9A4A2665B5D9
  - Remaining 312 bits are scrambled with PRBS13, using generator polynomial $X^{13} + X^{12} + X^2 + X + 1$, seed reset to 0xCCC for each pad fragment):
    - 38 byte Message field – Start of scrambling with PRBS
      - 8 bit message index (8 bit counter 0 to 255)
      - 8 bit message type (see slides 4 & 5)
      - 36 bytes message content
    - 1 byte CRC8 (calculated on previous 38 bytes) – polynomial is $X^8+X^5+X^4+1$
- **The 38-bytes message field (details to be specified) can be used to convey link and signal-related information, such as receiver state, channel pulse response, FEC stats, etc**

From patra_3dj_01b_2303



Continuous 48-bit FS format in patra_3dj_01b_2303

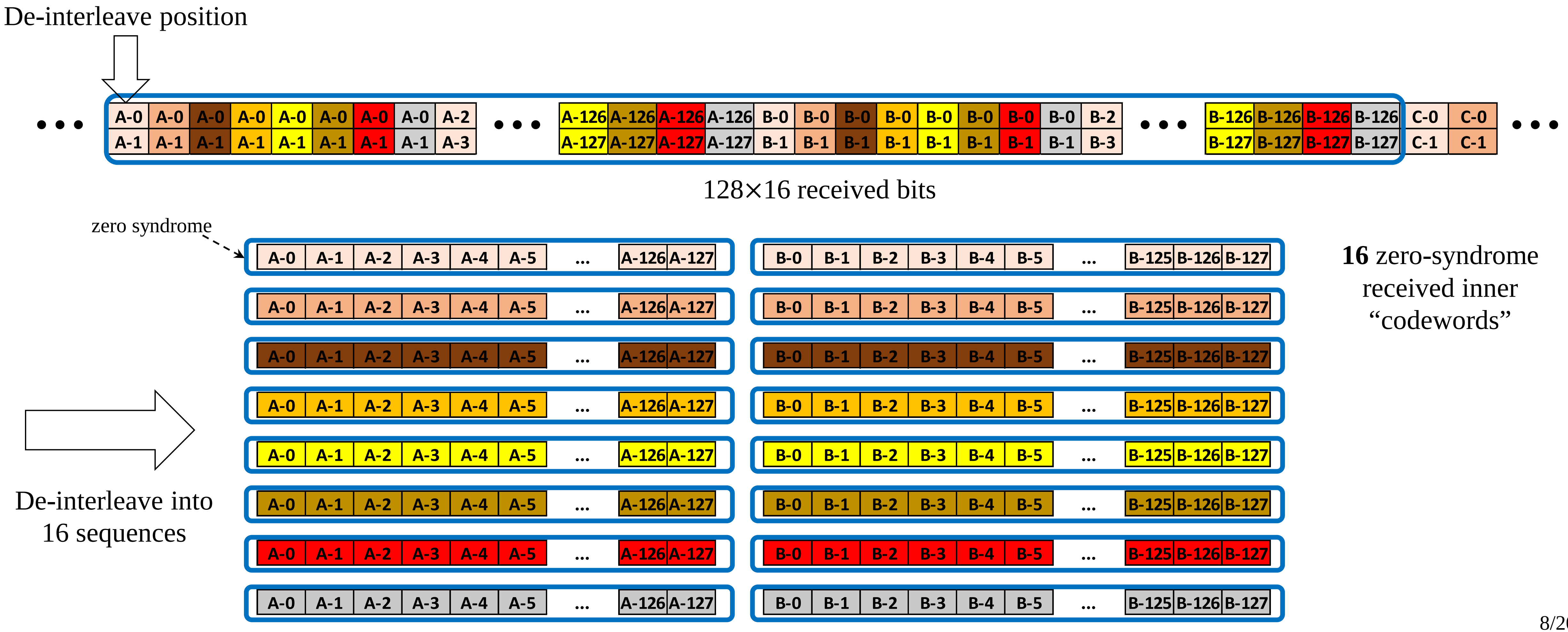


Proposed 24-bit FS + 24-bit FS format

*It can easily be extended for the Simplified Pad Insertion in rechtman_3dj_01a_2305

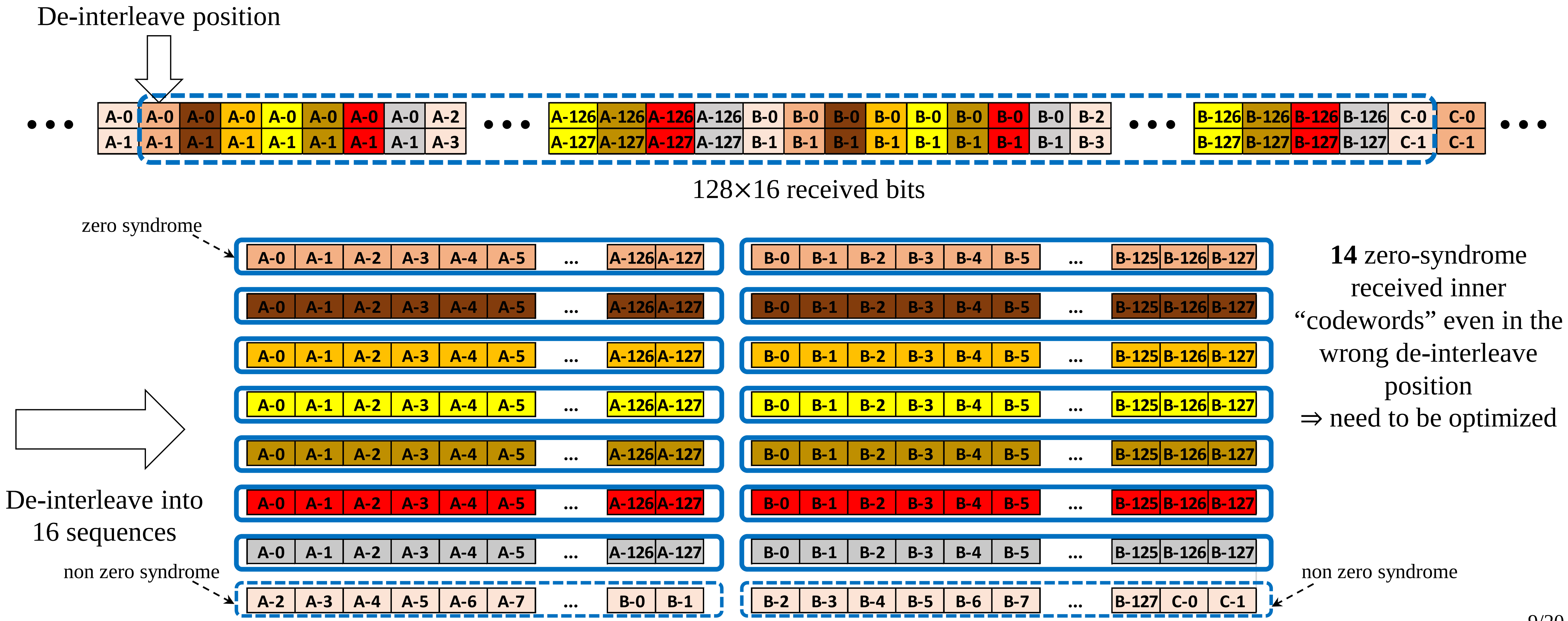
The Search & Test synchronization: at first glance (1/2)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - At first glance, an inner codeword boundary state machine may search for a pre-defined number ***T*** of zero-syndrome received inner “codewords” in a window of 128×***W*** received bits ⇒ may not work
  - Here, take ***W*** =16 and error-free received bits as example.



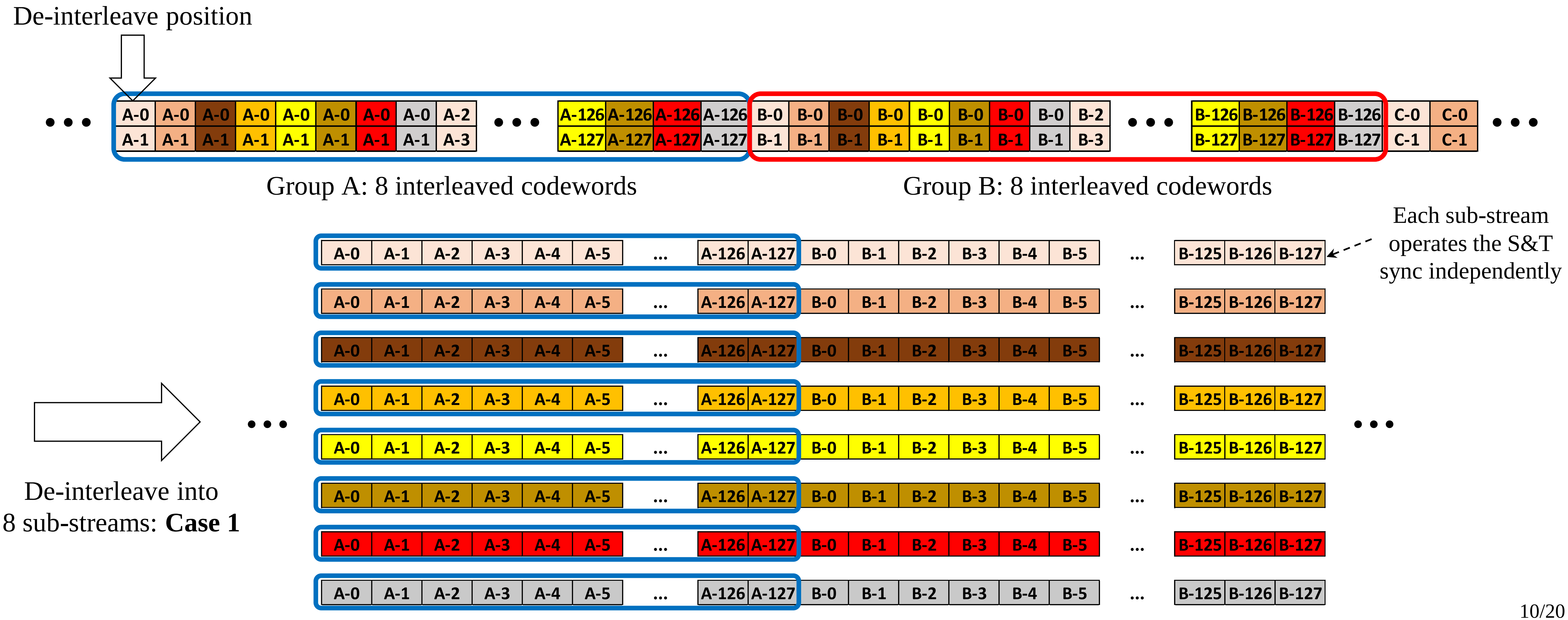
The Search & Test synchronization: at first glance (2/2)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - At first glance, an inner codeword boundary state machine may search for a pre-defined number ***T*** of zero-syndrome received inner "codewords" in a window of 128×***W*** received bits ⇒ may not work
  - Here, take ***W*** =16 and error-free received bits as example.



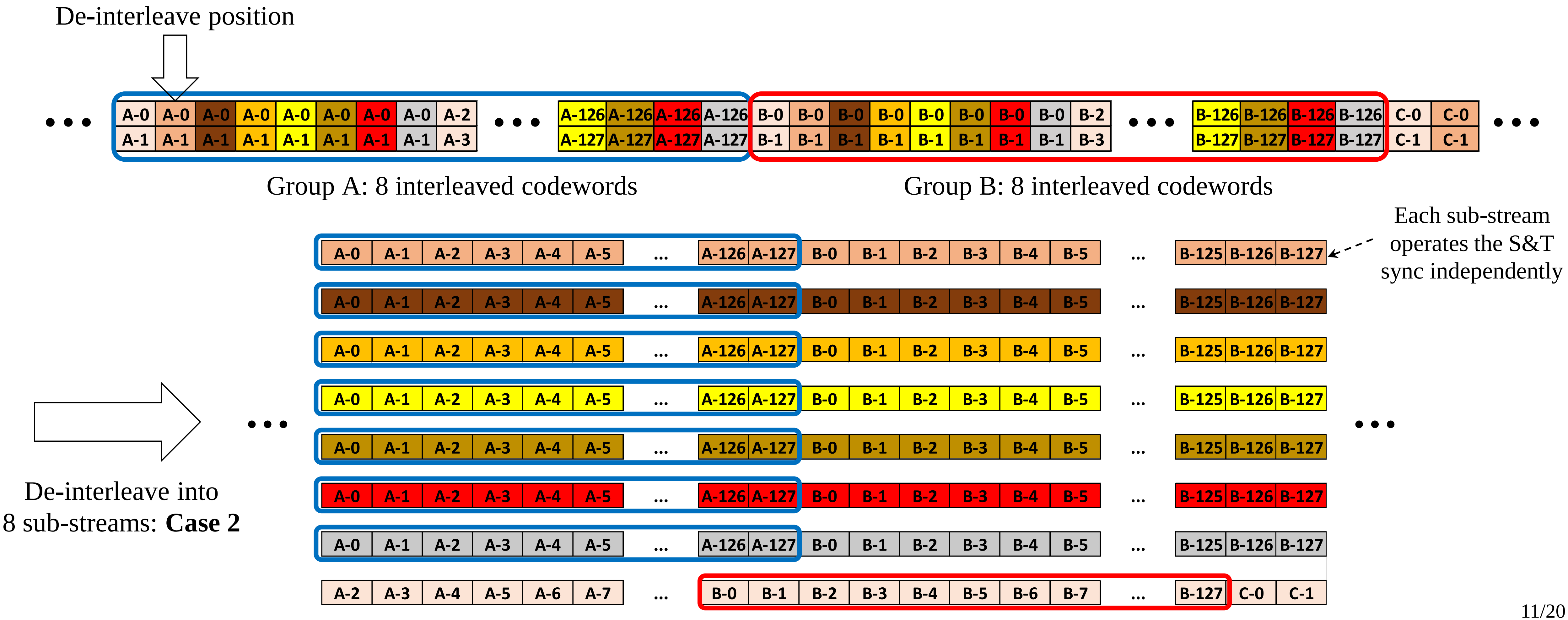
Potential Search & Test synchronization method (1/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



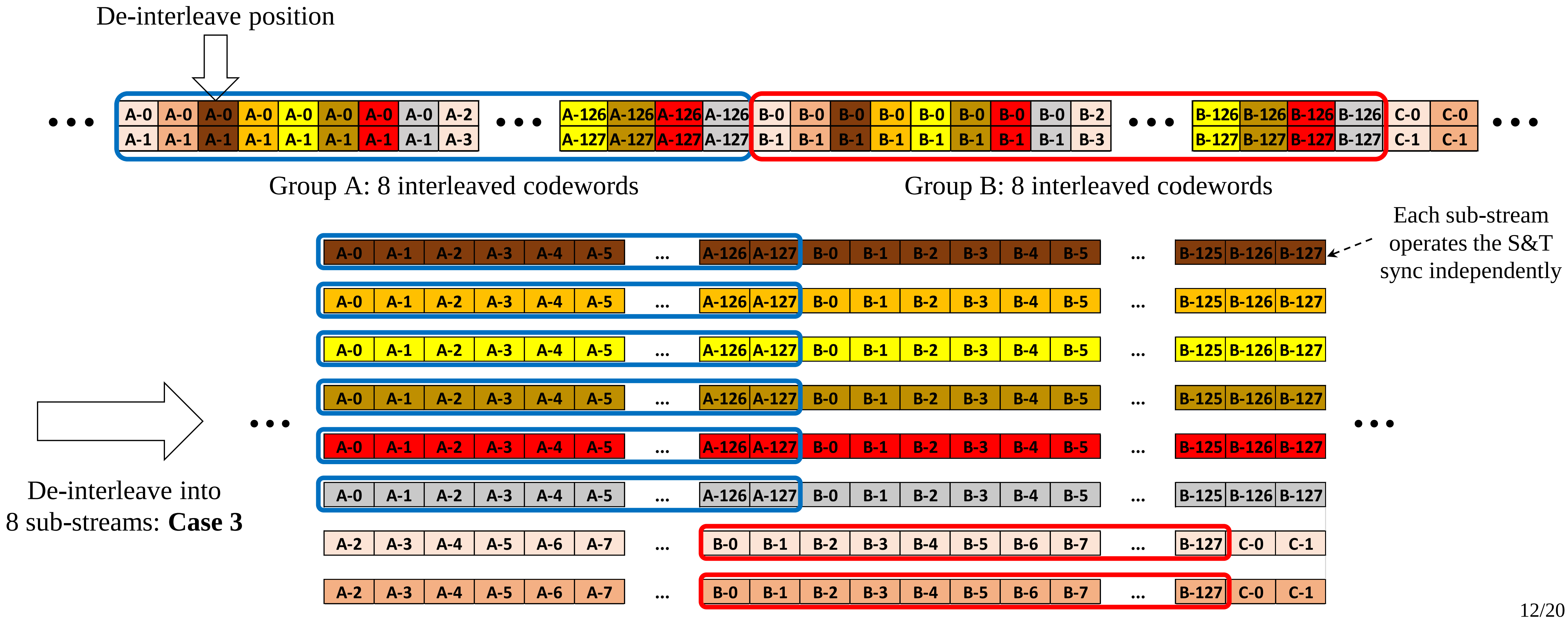
Potential Search & Test synchronization method (2/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



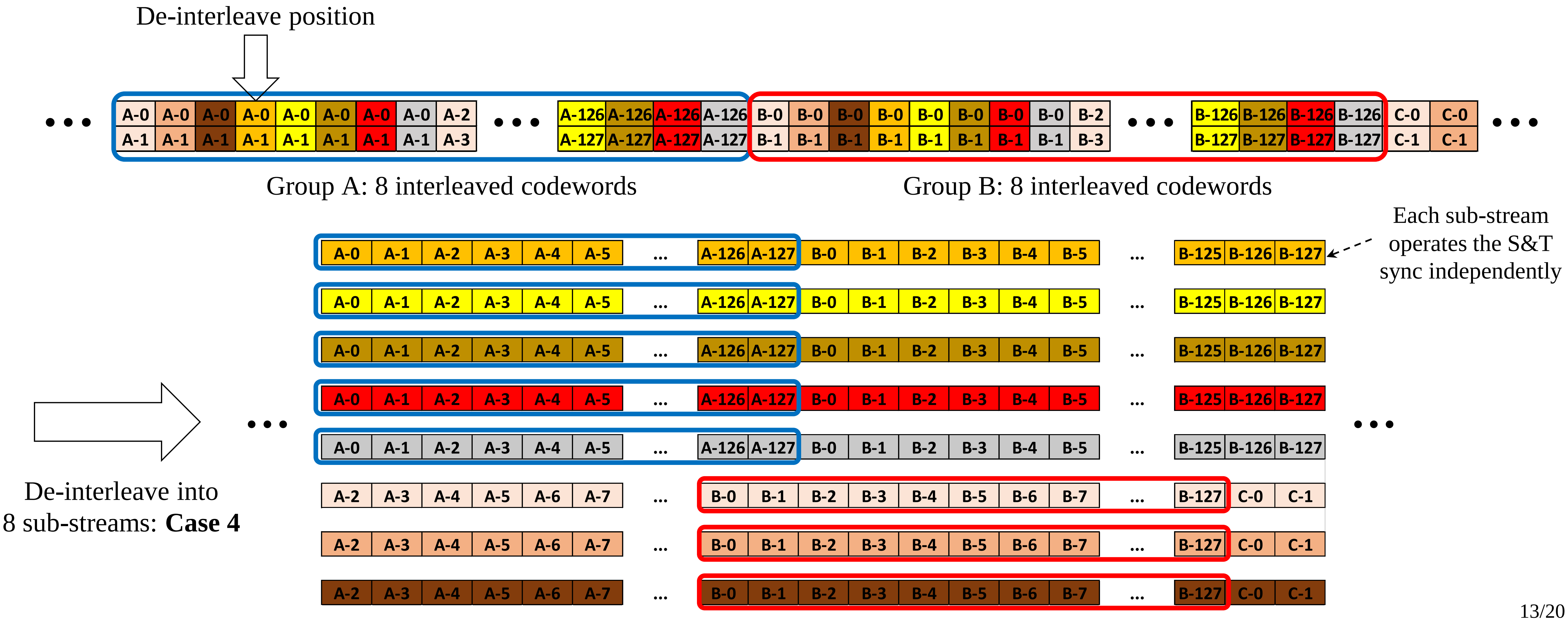
Potential Search & Test synchronization method (3/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



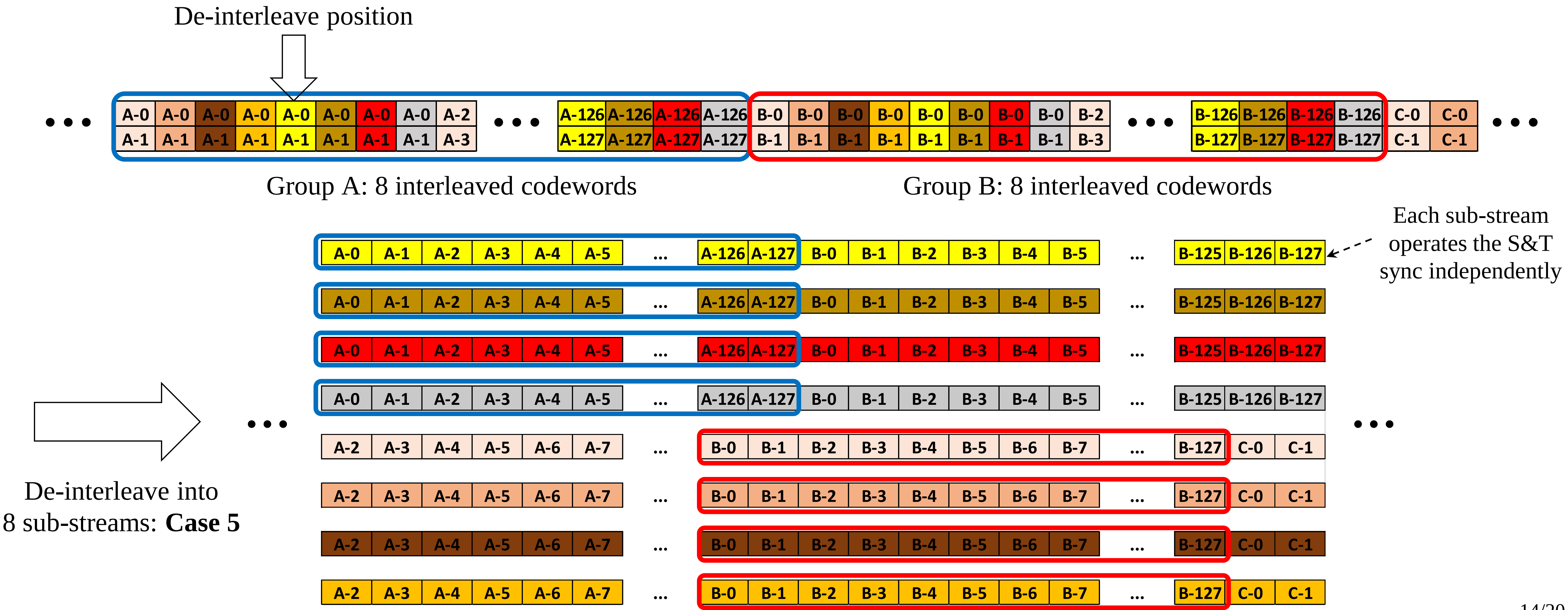
Potential Search & Test synchronization method (4/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



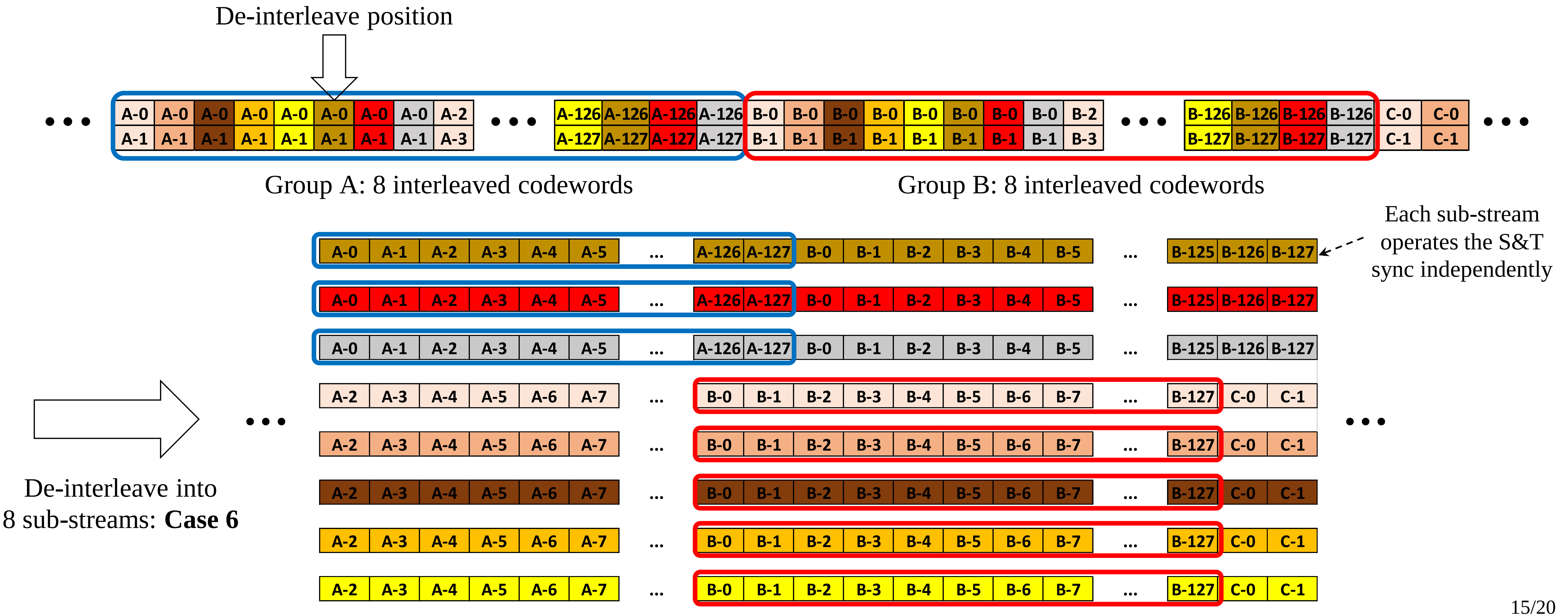
Potential Search & Test synchronization method (5/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



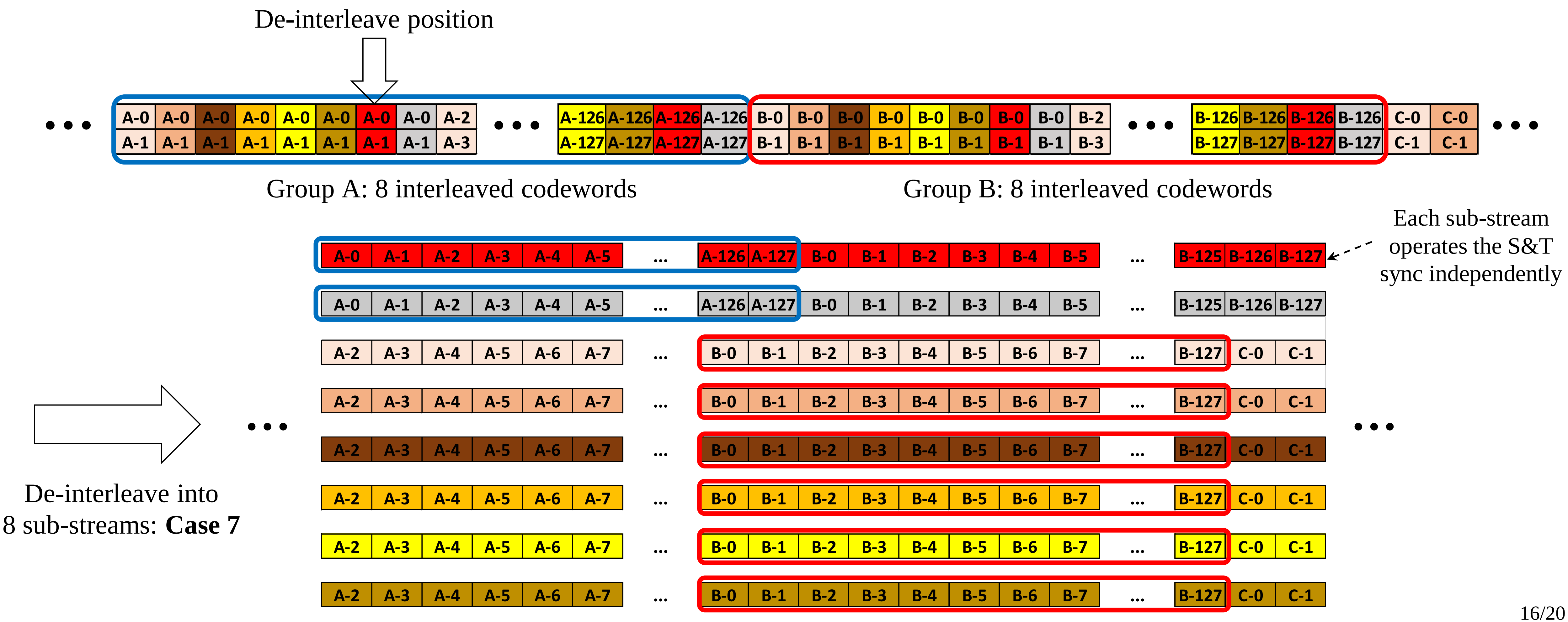
Potential Search & Test synchronization method (6/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



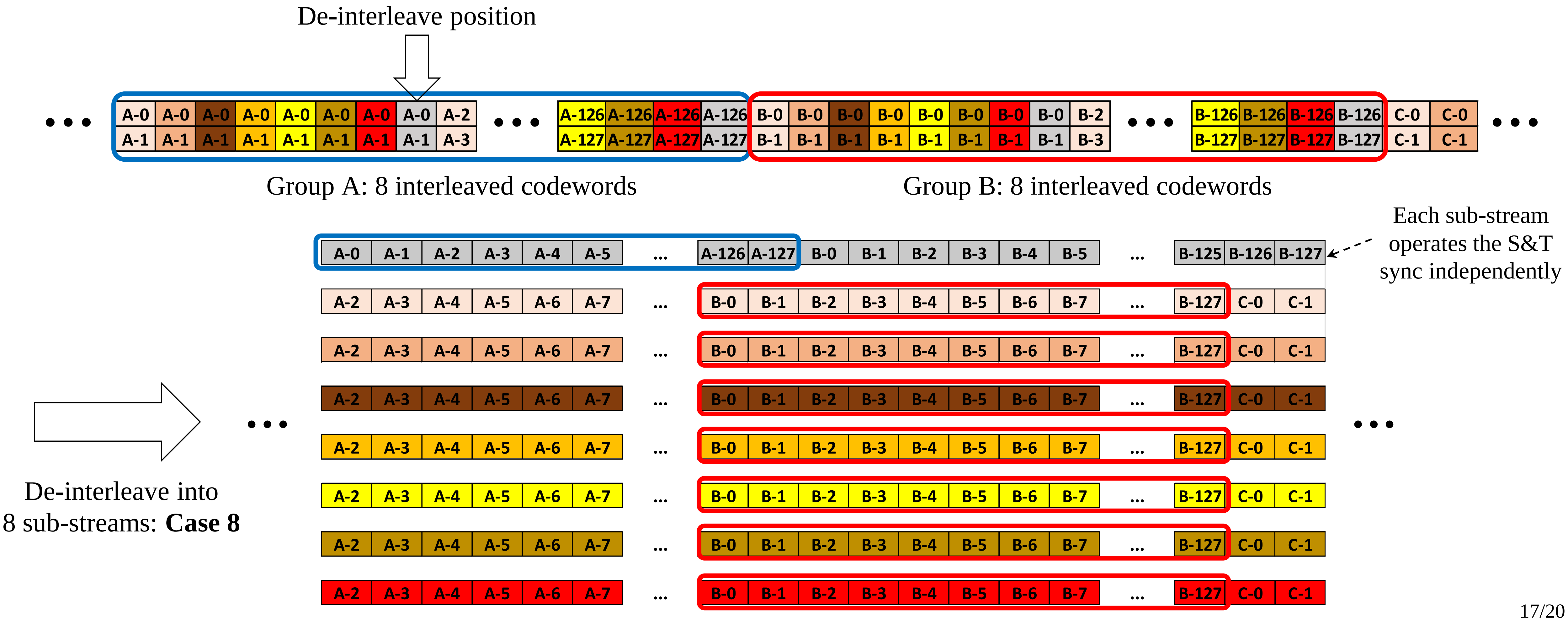
Potential Search & Test synchronization method (7/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received "codewords" in a window of 128×***W*** received bits



Potential Search & Test synchronization method (8/8)

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Need to take into consideration the effect of the 8-way Hamming codeword interleaver
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits



Comments on the Search & Test synchronization

- The inner codeword boundary can be identified by using the Hamming syndrome checking
  - Traditional S&T synchronization method may use an inner codeword boundary state machine to search for a pre-defined number ***T*** of zero-syndrome received inner “codewords” in a window of 128×***W*** received bits, which is operated on the received bit stream
  - Above traditional S&T synchronization method may not work, need to be improved
  - The effect of the 8-way Hamming codeword interleaver need to be taken into consideration
- One potential S&T synchronization method is provided to identify the Hamming codeword boundary
  - In the receiver side, the received bit stream is de-interleaved into eight sub-streams
  - Each S&T state machine on a sub-stream searches for ***T*** zero-syndrome received “codewords” in a window of 128×***W*** received bits
  - The FS lock is still required to identify the location of padding bits after the above Search & Test synchronization

Summary and Conclusions

- We presented detailed framing sequence (FS) lock process for future draft document
  - Consider the 48-bit FS (0x9A, 0x4A, 0x26, 0x65, 0xB5, 0xD9) same as the common marker (CM) portion of 200G/400G/800G PCS AM
  - Include detailed state diagram of Framing sequence lock
  - The FS lock can identify both the Hamming codeword boundary and the location of padding bits
- We proposed an improved FS format
  - Mimic the organization of the 200G/400G/800G PCS AM for hardware reuse purpose
  - The 48-bit FS consist of two groups, each with three bytes, and there is a one byte gap between the two groups
- We discussed and analyze the Search & Test synchronization method
  - The proposed Search & Test synchronization can be used to identify the Hamming codeword boundary
  - The FS lock is still required to identify the location of padding bits after the Search & Test synchronization

Thank you