256b/257b Transcoding for 100 Gb/s Backplane and Copper Cable

IEEE 802.3bj

March 12-16, 2012, Waikoloa, HI

Roy Cideciyan - IBM

Supporters

Zhongfeng Wang – Broadcom
Stephen Bates – PMC-Sierra
Mark Gustlin – Xilinx
Myles Kimmitt – Emulex
Jeff Slavick – Avago Technologies
Sudeep Bhoja – Broadcom
Matt Brown – AppliedMicro
Oren Sela – Mellanox
Pravin Patel – IBM
Martin Langhammer – Altera
John Ewen – IBM

Adam Healey – LSI
Mounir Meghelli – IBM
Dariush Dabiri – AppliedMicro
Vasudevan Parthasarathy – Broadcom
Troy Beukema – IBM
Arthur Marris – Cadence
Andre Szczepanek – Inphi
Barry Barnett – IBM
Daniel Dove – AppliedMicro
Frank Chang – Vitesse
Stephen Trowbridge – Alcatel-Lucent

Introduction

- 512b-based transcoding (TC)
  - 512b/513b TC scheme for 40 Gb/s ethernet: *trowbridge_01_0707.pdf*
  - 512b/513b TC scheme for 100 Gb/s ethernet backplane and copper cable: *cideciyan_01a_0911.pdf*
  - 512b/514b TC scheme for 100 Gb/s ethernet: a) *Teshima et al. "Bit-Error-Tolerant (512*N)B/(513*N+1)B Code for 40Gb/s and 100Gb/s Ethernet Transport", INFOCOM 2008*, b) *cideciyan_01a_1111.pdf*, c) *wang_01a_1111.pdf*, d) *gustlin_01_0112.pdf* and e) *brown_01_0112.pdf*
- The granularity of 256b-based TC simplifies processing of alignment markers: *gustlin_01_0312.pdf*
- 256b/257b TC with reshuffling: FEC Proposal for NRZ-Based 100G-KR Systems, Broadcom, Feb. 8, 2012.
- A proposal is shown for 256b/257b transcoding w/o reshuffling that can be used for both NRZ and PAM4 signaling

64b/66b Coding in 100GBASE-R

- 64b/66b coding used in 100GBASE-R (IEEE 802.3ba-2010, Clause 82)
  - 1 type of data block (DB) with 2-bit header **01**
  - 11 types of control blocks (CB) with 2-bit header **10** where the first byte of the payload is rate-4/8 encoded (Hamming distance=4) **8-bit block type field** indicating the type of control block format

	Input Data	Sync (Bit Position: 0 1)	Block Payload (Bit Position: 2 … 65)							
DB	Data Block Format:									
	$D_0 D_1 D_2 D_3 D_4 D_5 D_6 D_7$	01	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
CB	Control Block Formats:		Block Type Field							
1	$C_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7$	10	0x1E	C_0 C_1	C_2	C_3	C_4	C_5	C_6	C_7
2	$S_0 D_1 D_2 D_3 D_4 D_5 D_6 D_7$	10	0x78	D_1	D_2	D_3	D_4	D_5	D_6	D_7
3	$O_0 D_1 D_2 D_3 Z_4 Z_5 Z_6 Z_7$	10	0x4B	D_1	D_2	D_3	O_0	0x000_0000		
4	$T_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7$	10	0x87	C_1	C_2	C_3	C_4	C_5	C_6	C_7
5	$D_0 T_1 C_2 C_3 C_4 C_5 C_6 C_7$	10	0x99	D_0	C_2	C_3	C_4	C_5	C_6	C_7
6	$D_0 D_1 T_2 C_3 C_4 C_5 C_6 C_7$	10	0xAA	D_0	D_1	C_3	C_4	C_5	C_6	C_7
7	$D_0 D_1 D_2 T_3 C_4 C_5 C_6 C_7$	10	0xB4	D_0	D_1	D_2	C_4	C_5	C_6	C_7
8	$D_0 D_1 D_2 D_3 T_4 C_5 C_6 C_7$	10	0xCC	D_0	D_1	D_2	D_3	C_5	C_6	C_7
9	$D_0 D_1 D_2 D_3 D_4 T_5 C_6 C_7$	10	0xD2	D_0	D_1	D_2	D_3	D_4	C_6	C_7
10	$D_0 D_1 D_2 D_3 D_4 D_5 T_6 C_7$	10	0xE1	D_0	D_1	D_2	D_3	D_4	D_5	C_7
11	$D_0 D_1 D_2 D_3 D_4 D_5 D_6 T_7$	10	0xFF	D_0	D_1	D_2	D_3	D_4	D_5	D_6

BTF

Figure 82–5—64B/66B block formats

256b/257b Transcoding with Reshuffling

Transcoding Structure

- 256B/257B transcoding structure is similar to 512B/514B transcoding [2] with 2 major differences, 1) smaller TC block size, 2) only one sync bit.
- In the following figure, F is 1-b flag to indicate the next 64-b block is a control block (F=1) or data block (F=0).
- CBT is 4-bit encoding control block type.
- 3-b POS field consists of 2 bits for original position of the control block and 1-b parity information. This 1-b parity is chosen as parity of 1-b flag, 2-b position index, and 4-b CBT.

01	D0	D1	D2	D3	D4	D5	D6	D7
10	Ca	rest 7 bytes						
01	D0	D1	D2	D3	D4	D5	D6	D7
10	Cb	rest 7 bytes						

F	POS	CBT

0	Ta	rest 7 bytes						
	Tb	rest 7 bytes						
	D0	D1	D2	D3	D4	D5	D6	D7
	D0	D1	D2	D3	D4	D5	D6	D7

[2] M. Teshima, etc, "Bit-Error-Tolerant (512*N)B/(513*N+1)B Code for 10Gb/s and 100Gb/s Ethernet Transport," IEEE Infocom Workshops 2008.

BROADCOM. Connecting everything

Presentation by Zhongfeng Wang, FEC Proposal for NRZ-Based 100G-KR Systems, Broadcom, Feb. 8, 2012.

256b/257b Transcoding Proposal w/o Reshuffling

Benefits of 256b/257b TC vs. 512b/514b TC

gustlin_01_0112.pdf and gustlin_01_0312.pdf

FEC Code RS(n, k, t, m)	Transcoding (TC)	Effective Gain for BER = 10^{-15}	Overall Latency *	Total Area (40nm gates) *	Total Power *
RS(528, 514, 7, 10)	256b/257b	4.87 dB	94.3 ns	244k	90 mW
RS(528, 514, 7, 10)	512b/514b	4.87 dB	99.4 ns	285k	105 mW

- The granularity of 256b/257b TC simplifies processing of alignment markers because the total number of 20 alignment markers is divisible by 4 (# of control/data blocks in 256b-based TC) but not by 8 (# of control/data blocks in 512b-based TC): *gustlin_01_0312.pdf*
- RS(528,514), t=7, m=10 code provides the same error-rate performance and the same coding gain of 4.87 dB in case of 256b/257b or 512b/514b TC scheme
- 5% to 7.5% reduction in overall latency associated with transcoding and FEC
- 15% reduction in total gate complexity
- 15% reduction in total power dissipation

* Zhongfeng Wang computed overall latency, total gate complexity and power dissipation for a nominal design

Data Blocks and Control Blocks

- 256b/257b transcoding converts four incoming 66-bit blocks into one 257-bit block
- Four incoming 66-bit blocks that have to be transcoded may be data blocks (DB) with 2-bit header **01** or control blocks (CB) with 2-bit header **10**
- Data block #i with 64-bit block payload DBi(64) where bits are sent from left to right

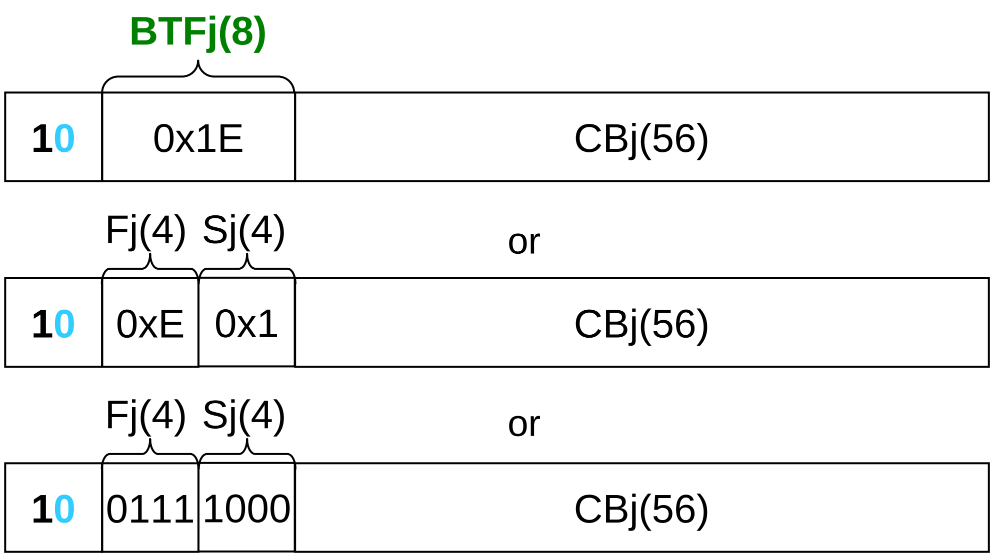
01	DBi(64)

- Control block #j with 8-bit block type field **BTFj(8)** and 56-bit control information CBj(56) where bits are sent from left to right

10	**BTFj(8)**	CBj(56)

Block Type Field

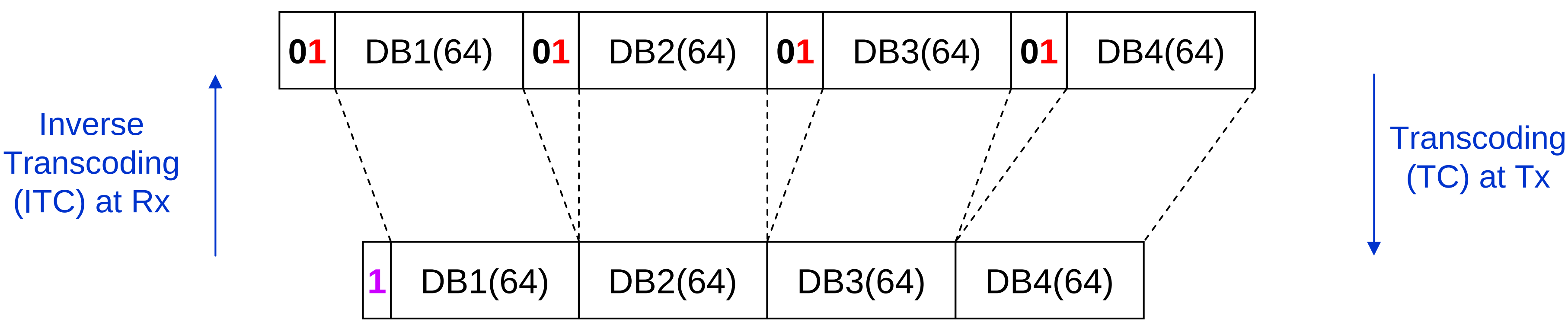
- According to clause 82.2.3.1 the LSB of the block type field (BTF) represented as a hexadecimal value is the first transmitted bit. For example, the block type field 0x1E is sent from left to right as 01111000.
- BTFj(8)=BTFj<7:0> where BTFj<0> is the first transmitted bit. We represent BTFj(8) as the concatenation of the first nibble Fj(4)=BTFj<3:0> and the second nibble Sj(4)=BTFj<7:4>. For example, for BTFj(8)=0x1E, we obtain Fj(4)=0xE sent from left to right as 0111 and Sj(4)=0x1 sent from left to right as 1000.



256b/257b Transcoding for DB-Only Payload

- *Step* 1: *Delete* **01** header bits from all DBs
- *Step* 2: *Insert* header bit 1 indicating 256-bit payload of transcoded block contains only block payloads of DBs
- DBk(64), k=1, 2, 3 and 4, indicates 64-bit block payload associated with DB #k
- Transcoded blocks are sent from left to right (leftmost bit first, rightmost bit last). Header bit of transcoded block 1 is sent first.

Only Case: DB #1, DB #2, DB #3 and DB #4



No reshuffling of the order of data blocks after transcoding

256b/257b Transcoding w/ at least 1 CB

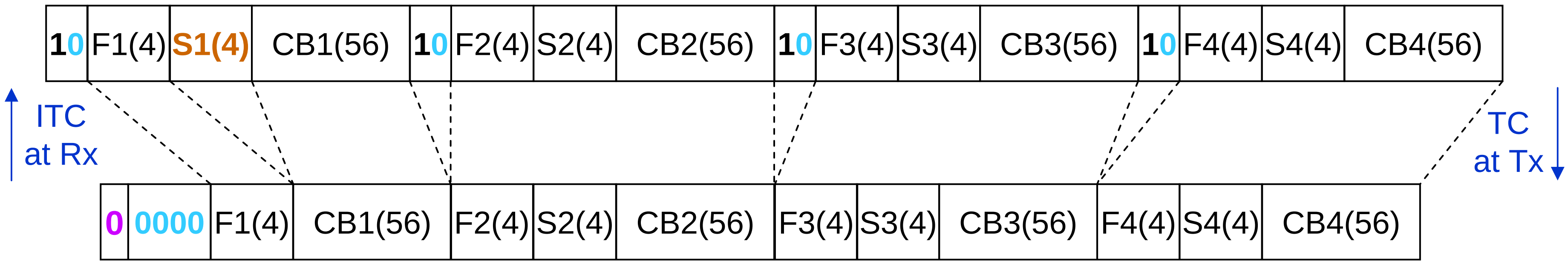
- *Step* 1: *Delete* **01** header bits of all DBs and **10** header bits of all CBs.
- *Step* 2: *Insert* header bit 0 (sent first) indicating 256-bit payload of transcoded block contains at least 1 CB
- *Step* 3: *Delete* the second 4-bit nibble in the first byte of the first control block in a transcoded block. The first 4-bit nibble in the first byte of the first control block indicating the type of the first control block is kept. There are a total of 11 CB-type indicators without accounting for alignment markers (AM).
- *Step* 4: *Insert* 4-bit pattern x1 x2 x3 x4 following header bit 0 where xi=0 indicates that i-th block is control block CBi whereas xi=1 indicates that i-th block is data block DBi.
- 64-bit data in all second, third and fourth control blocks in a transcoded block are kept unchanged and therefore first byte of second, third and fourth control blocks indicating block type field is protected by distance-4 Hamming code.

No reshuffling of the order of data/control blocks after transcoding

256b/257b Transcoding w/ at least 1 CB (cont.)

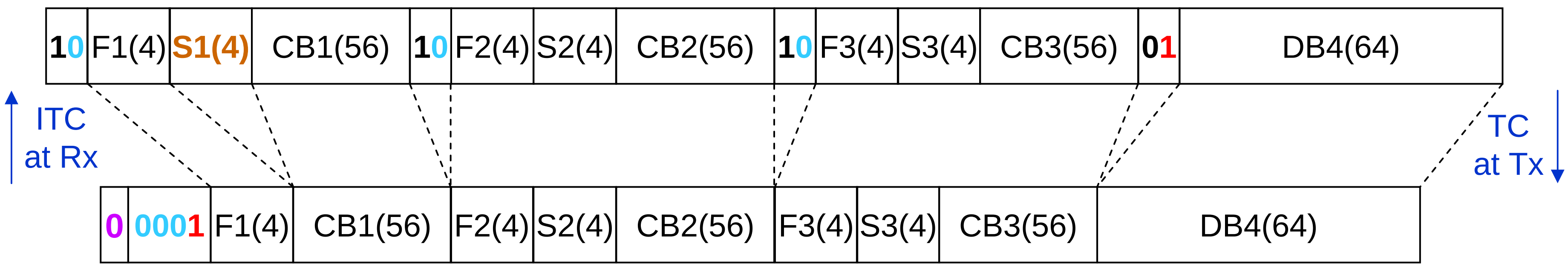
- Transcoded blocks are sent from left to right (leftmost bit first, rightmost bit last). Header bit of transcoded block 0 is sent first.
- Notation
  - 1st block may be a data block DB1(64) or control block F1(4), S1(4) and CB1(56)
  - 2nd block may be a data block DB2(64) or control block F2(4), S2(4) and CB2(56)
  - 3rd block may be a data block DB3(64) or control block F3(4), S3(4) and CB3(56)
  - 4th block may be a data block DB4(64) or control block F4(4), S4(4) and CB4(56)

Case 1: CB #1, CB #2, CB #3 and CB #4

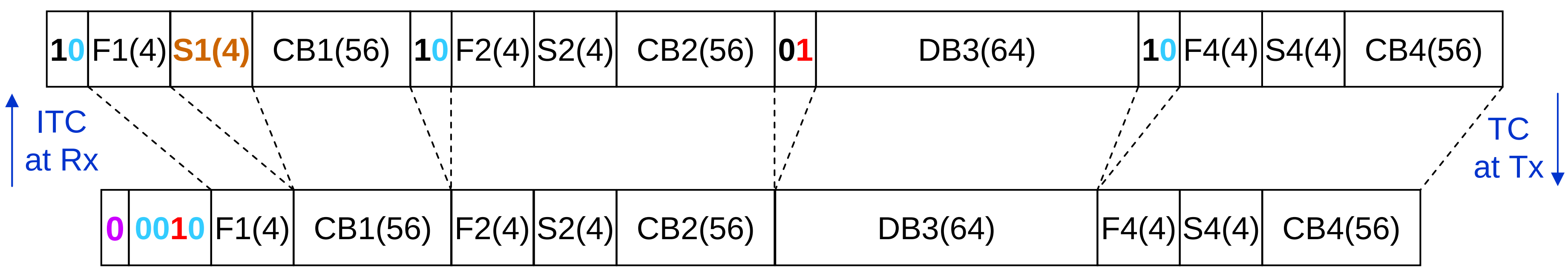


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 2: CB #1, CB #2, CB #3 and DB #4

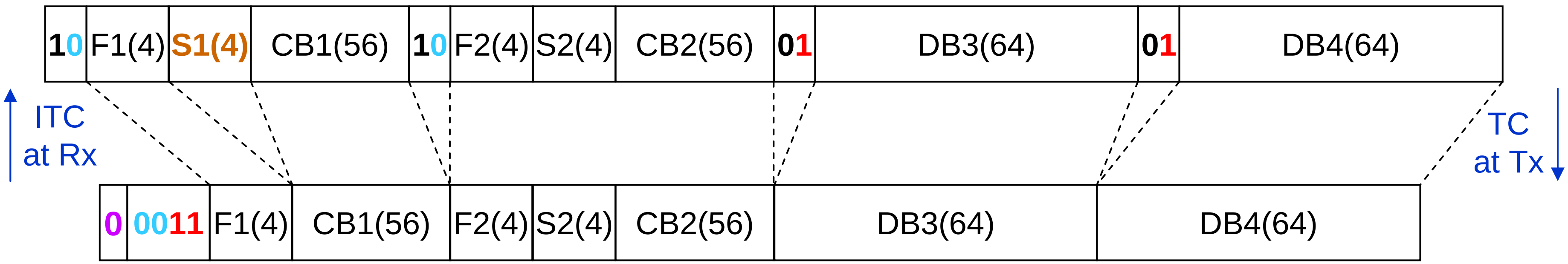


Case 3: CB #1, CB #2, DB #3 and CB #4

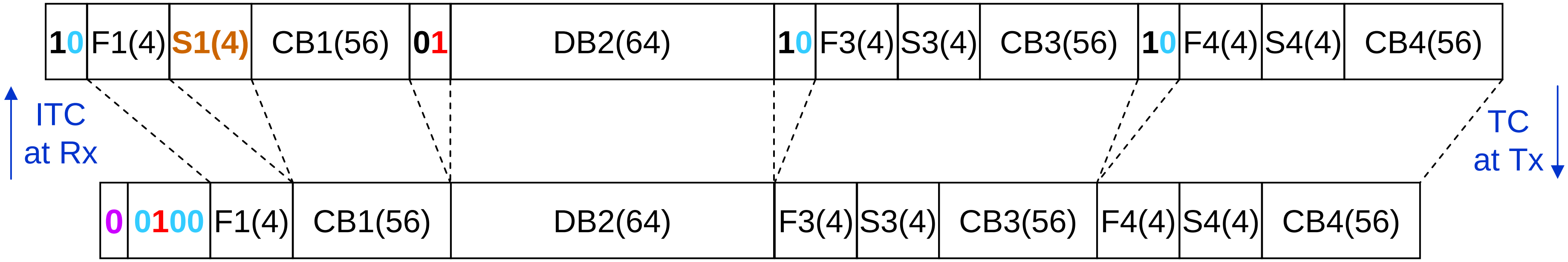


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 4: CB #1, CB #2, DB #3 and DB #4

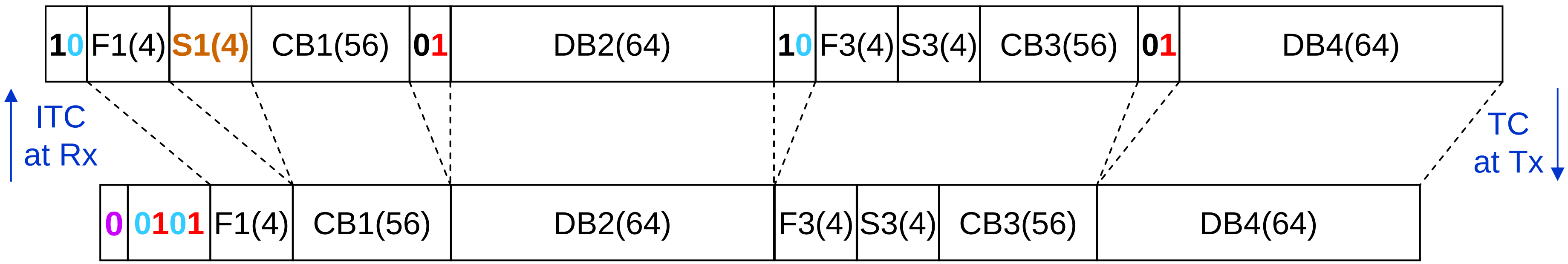


Case 5: CB #1, DB #2, CB #3 and CB #4

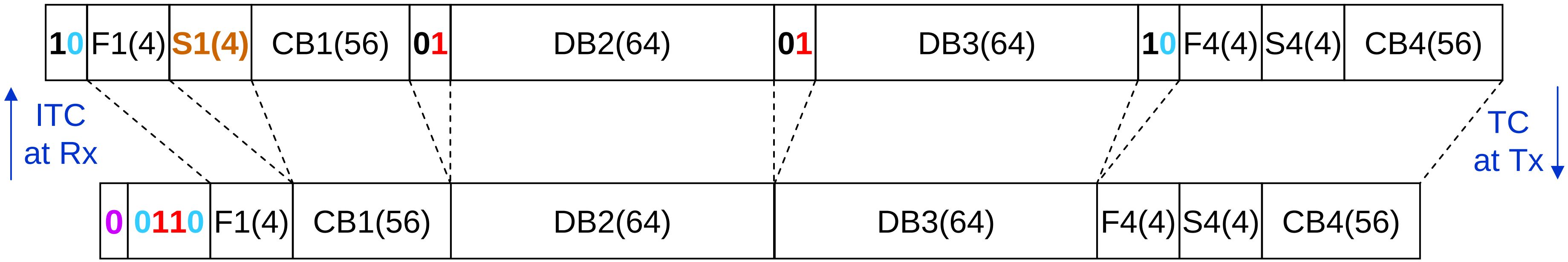


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 6: CB #1, DB #2, CB #3 and DB #4

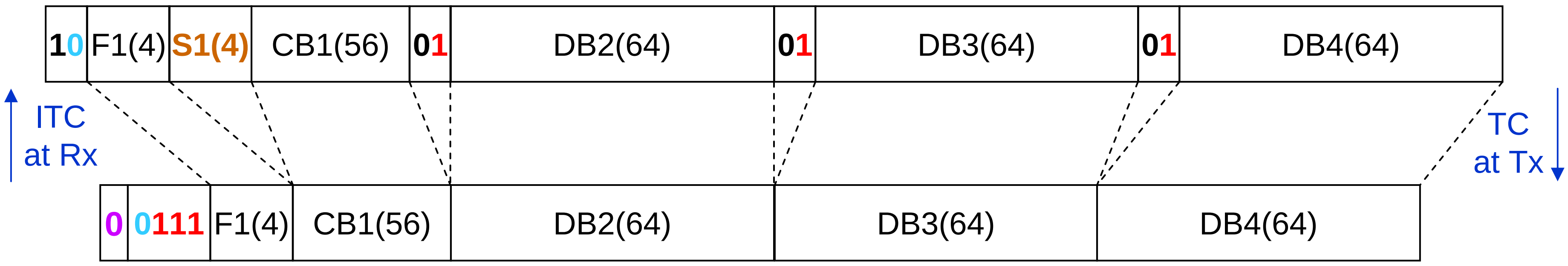


Case 7: CB #1, DB #2, DB #3 and CB #4

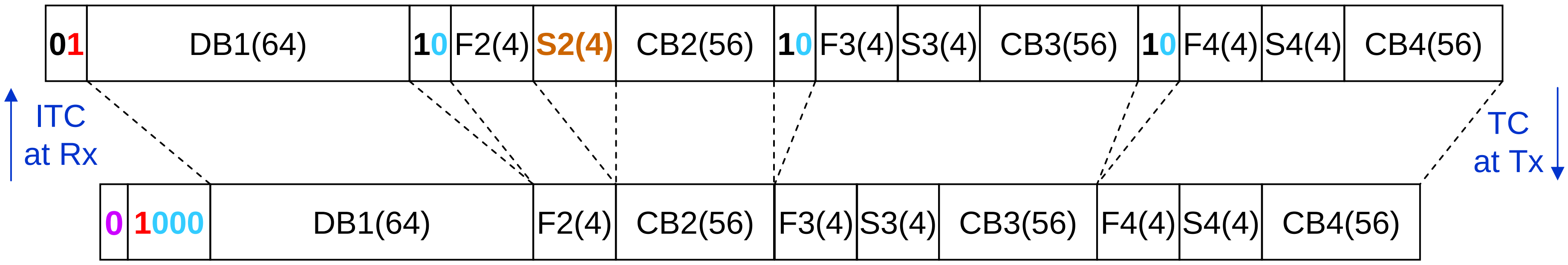


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 8: CB #1, DB #2, DB #3 and DB #4

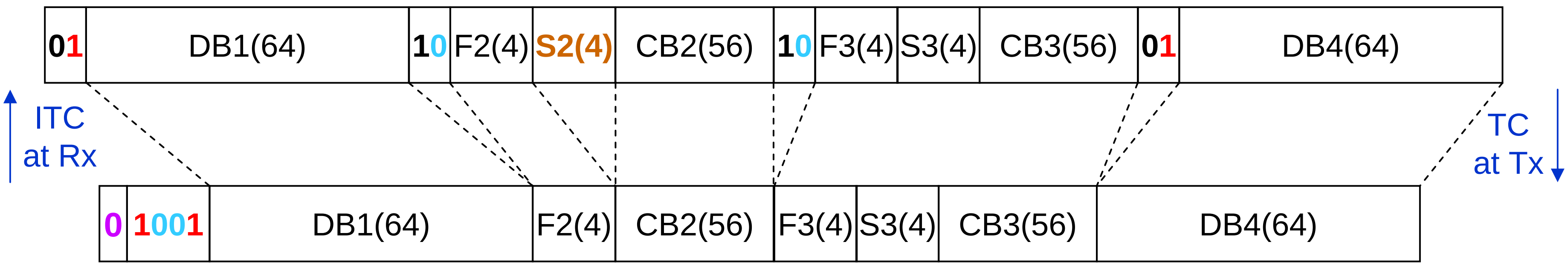


Case 9: DB #1, CB #2, CB #3 and CB #4

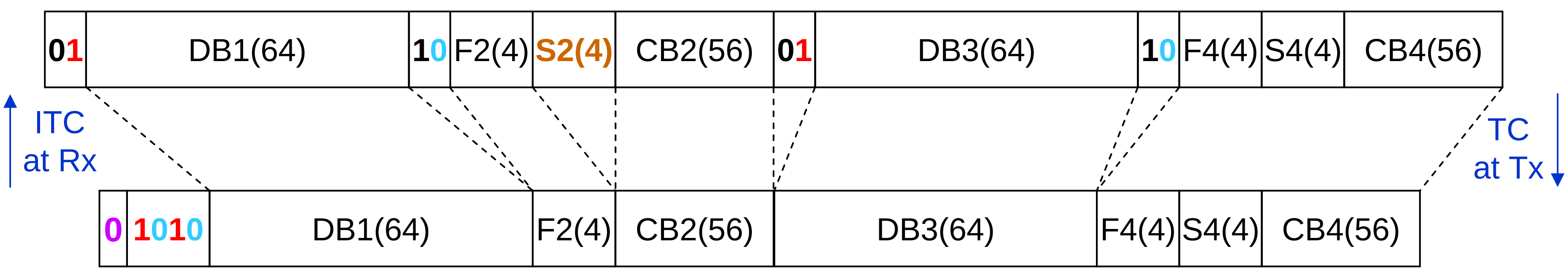


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 10: DB #1, CB #2, CB #3 and DB #4

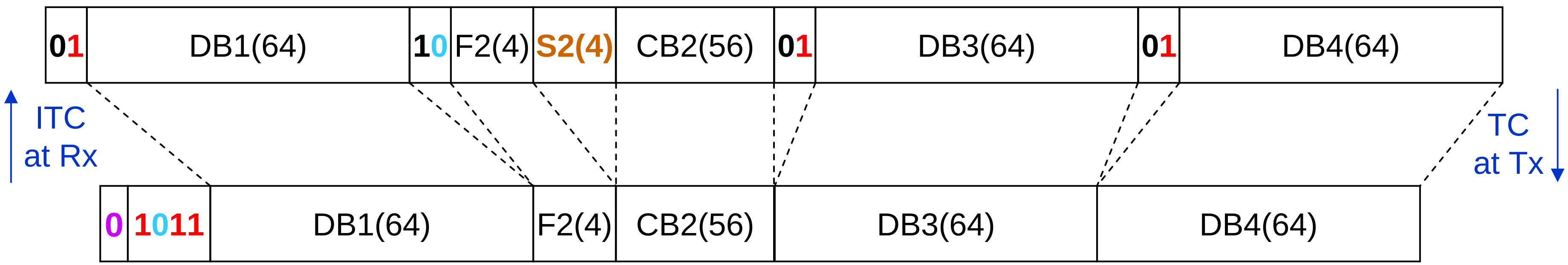


Case 11: DB #1, CB #2, DB #3 and CB #4

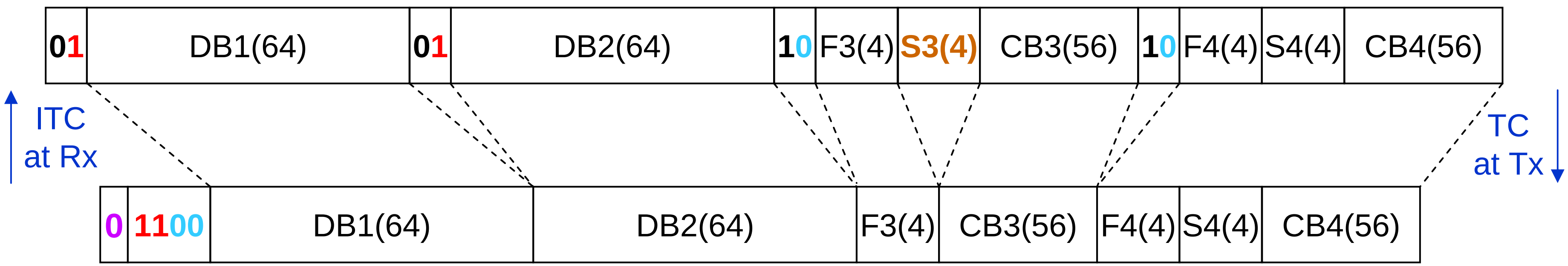


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 12: DB #1, CB #2, DB #3 and DB #4

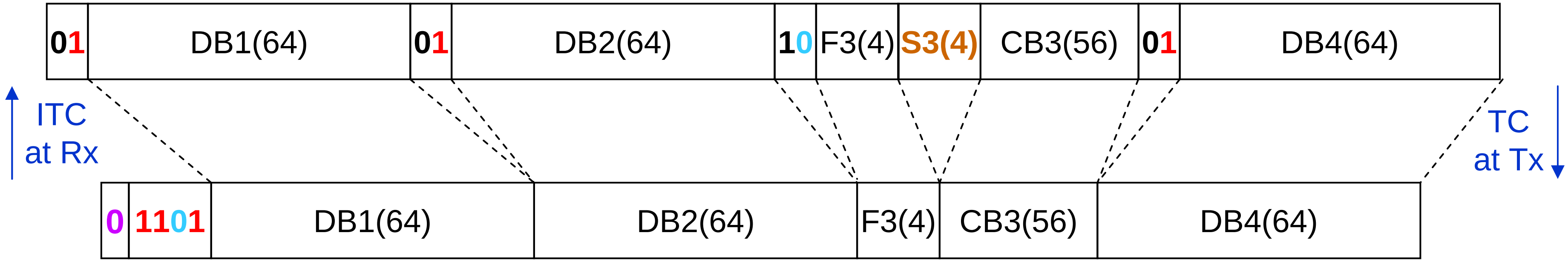


Case 13: DB #1, DB #2, CB #3 and CB #4

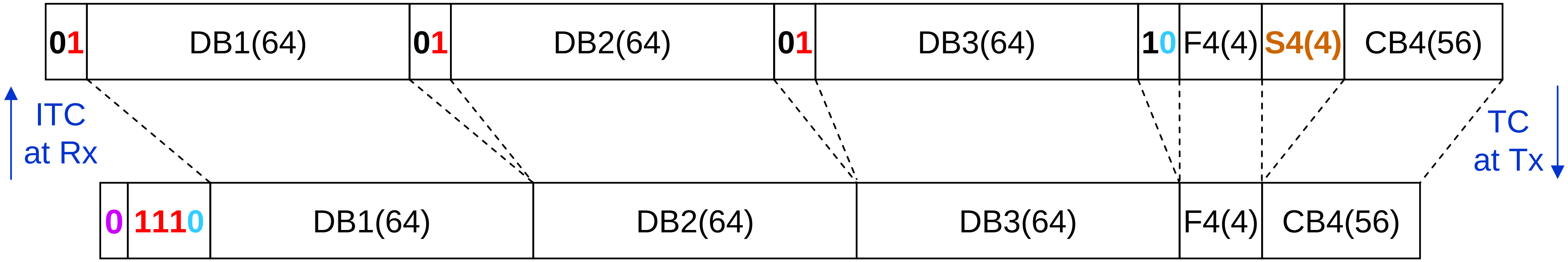


256b/257b Transcoding w/ at least 1 CB (cont.)

Case 14: DB #1, DB #2, CB #3 and DB #4



Case 15: DB #1, DB #2, DB #3 and CB #4



Detection of Invalid Transcoded Blocks

- Invalid transcoded blocks (ITB) can be detected during inverse transcoding
- *ITB condition* 1: Header bit 0 followed by x1=1, x2=1, x3=1, x4=1
- *ITB condition* 2: First 4-bit nibble (16 possible patterns) in first control block is one out of 16-11=5 invalid CB type indicators
- *ITB condition* 3: 8-bit BTF (256 possible patterns) of second, third and fourth CBs in transcoded block is one out of 256-11=245 illegal block type fields. In this case, the distance-4 Hamming code is used as an error detection code that is capable of detecting 245 out of 255 possible error patterns. An alternative would be to correct all single-bit errors in 8-bit BTF (256 possible patterns) of second, third and fourth CBs in transcoded block and to detect 256-11*9=157 out of 255 possible error patterns. In this case, the distance-4 Hamming code is used as a single-bit error correction code and as an error detection code that is capable of detecting 157 error patterns.
- *ITB condition* 4: Invalid bits in 56-bit control payload CBi(56) of control block CB #i. Some bits in 56-bit control payload are set to "0" as specified in clause 82.2.3.3 (see thin rectangles and 0x000_0000 in Figure 82-5). If these bits are received as "1" in a transcoded block, an invalid transcoded block can be indicated.

Thank You