Improved Transcoding Scheme for 40GBASE-T

Addressing Comment #403 to Draft 2.0

Zhongfeng Wang

Broadcom Corporation

IEEE P802.3bq, Task Force meeting, May 2015

everything

Supporters

- Peter Wu, Marvell Tech. Group

Introduction

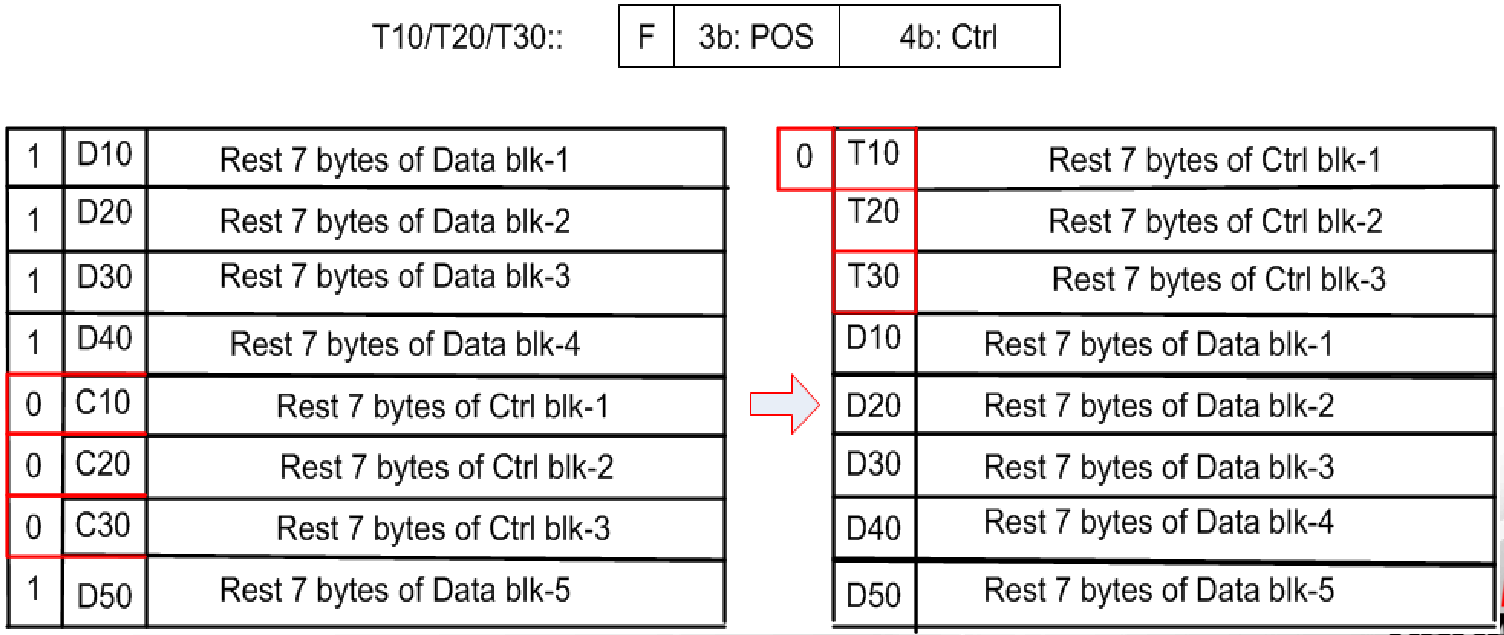
- **Consider the newly adopted 512b/513b transcoding scheme [1], this presentation aims to**
  - Reduce the logic complexity of transcoding and trans-decoding
  - Reduce power consumption of transcoding and trans-decoding
  - Reduce processing latency of trans-decoding

- **Basic Idea:**
  - Fully maintain the original data mapping for each individual byte of data during transcoding.
  - Reorder those transcoded data bytes to facilitate hardware reduction and processing latency reduction.

[1] http://www.ieee802.org/3/bq/public/jan15/langner_3bq_01a_0115.pdf

Review: 512b/513b Transcoding

- If all of 8 65-b blocks are data blocks, set the header bit as “1" while keeping the rest of 64 bits as they are for each 65-b block.
- If at least one of 8 65-b blocks is a control block, set the header bit as “0" and move all the control blocks to the beginning while changing 1st byte of each control block as T10 format shown below.



T10/T20/T30::

F	3b: POS	4b: Ctrl

1	D10	Rest 7 bytes of Data blk-1
1	D20	Rest 7 bytes of Data blk-2
1	D30	Rest 7 bytes of Data blk-3
1	D40	Rest 7 bytes of Data blk-4
0	C10	Rest 7 bytes of Ctrl blk-1
0	C20	Rest 7 bytes of Ctrl blk-2
0	C30	Rest 7 bytes of Ctrl blk-3
1	D50	Rest 7 bytes of Data blk-5

0	T10	Rest 7 bytes of Ctrl blk-1
	T20	Rest 7 bytes of Ctrl blk-2
	T30	Rest 7 bytes of Ctrl blk-3
	D10	Rest 7 bytes of Data blk-1
	D20	Rest 7 bytes of Data blk-2
	D30	Rest 7 bytes of Data blk-3
	D40	Rest 7 bytes of Data blk-4
	D50	Rest 7 bytes of Data blk-5

Step-1: Reduce Complexity

- To reduce implementation complexity, we ONLY swap (if needed) first byte for each 64-b block [2]. The data format for T10/T20/T30 remains the same as with conventional transcoding (TC).
- **With the new TC scheme, the MUXing logic is only associated with first 8 bits instead of 64 bits per 64-b block as in conventional design.**

T10/T20/T30::

F	3b: POS	4b: Ctrl

1	D10	Rest 7 bytes of Data blk-1
1	D20	Rest 7 bytes of Data blk-2
1	D30	Rest 7 bytes of Data blk-3
1	D40	Rest 7 bytes of Data blk-4
0	C10	Rest 7 bytes of Ctrl blk-1
0	C20	Rest 7 bytes of Ctrl blk-2
0	C30	Rest 7 bytes of Ctrl blk-3
1	D50	Rest 7 bytes of Data blk-5

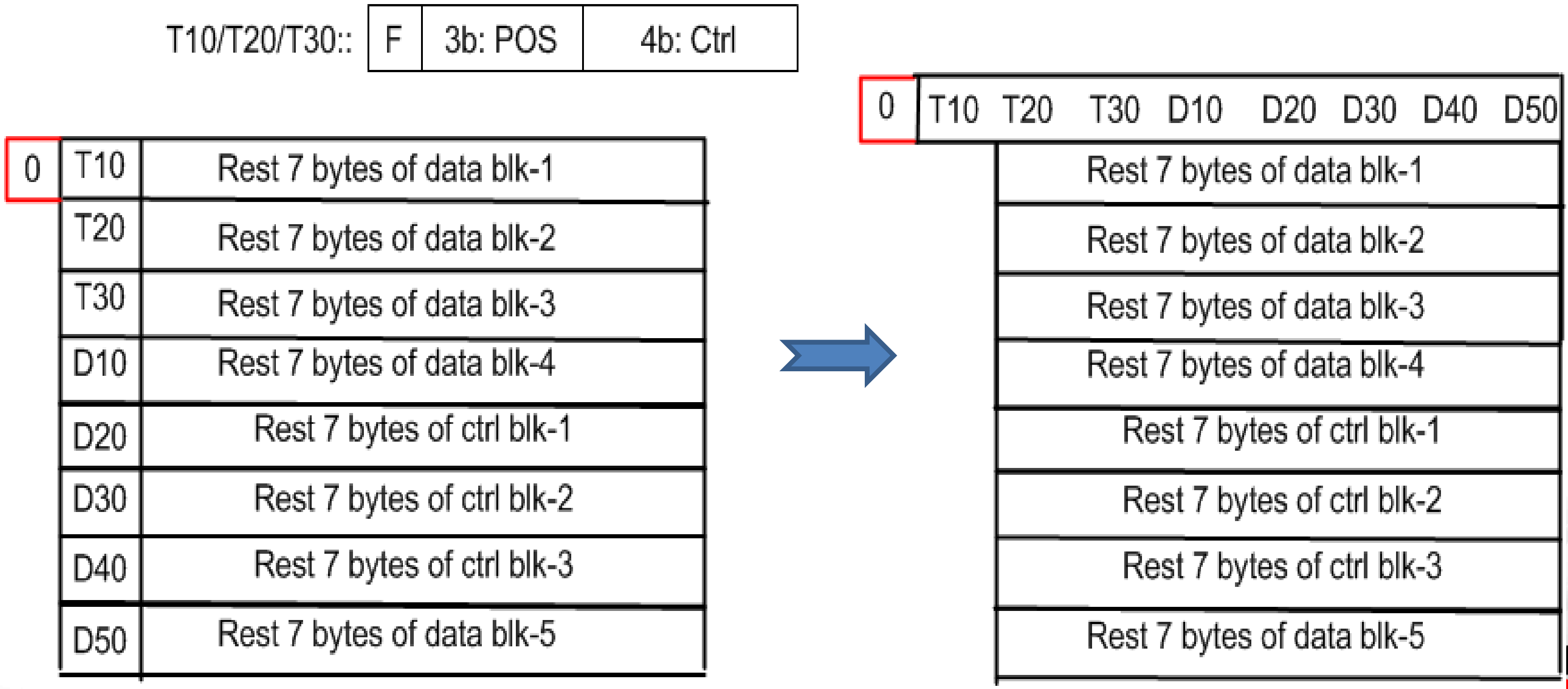
$\Rightarrow$

0	T10	Rest 7 bytes of Data blk-1
	T20	Rest 7 bytes of Data blk-2
	T30	Rest 7 bytes of Data blk-3
	D10	Rest 7 bytes of Data blk-4
	D20	Rest 7 bytes of Ctrl blk-1
	D30	Rest 7 bytes of Ctrl blk-2
	D40	Rest 7 bytes of Ctrl blk-3
	D50	Rest 7 bytes of Data blk-5

[2] http://www.ieee802.org/3/bs/public/adhoc/logic/dec02_14/wangz_01_1214_logic.pdf

Step-2: Reduce Latency

- To reduce processing latency, we transmit the first bytes of every 64-b blocks before sending out the rest 7 bytes of any 64-b block [2].
- In this way, the trans-decoding operation doesn't need to wait until the whole transcoded block of data is received.

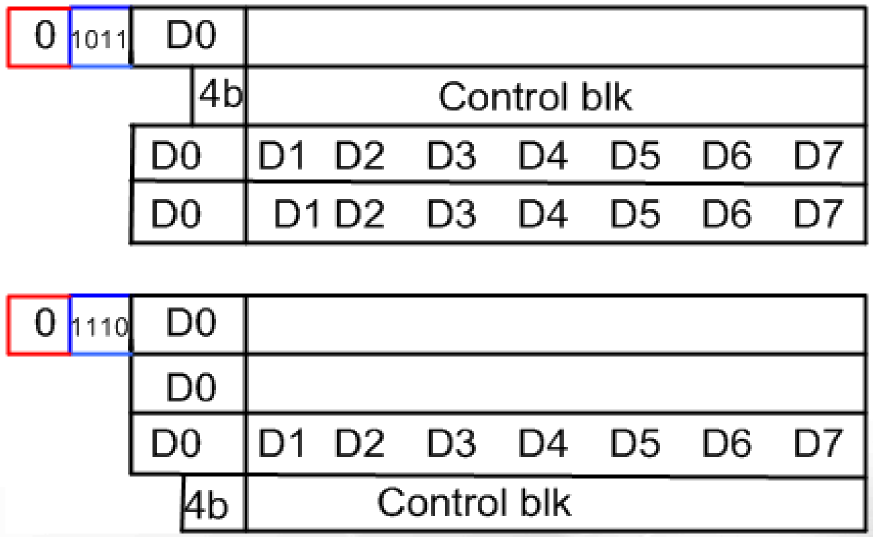


T10/T20/T30::
F
3b: POS
4b: Ctrl
0
T10
T20
T30
D10
D20
D30
D40
D50
Rest 7 bytes of data blk-1
Rest 7 bytes of data blk-2
Rest 7 bytes of data blk-3
Rest 7 bytes of data blk-4
Rest 7 bytes of ctrl blk-1
Rest 7 bytes of ctrl blk-2
Rest 7 bytes of ctrl blk-3
Rest 7 bytes of data blk-5

[2] http://www.ieee802.org/3/bs/public/adhoc/logic/dec02_14/wangz_01_1214_logic.pdf

Prior Work: 802.3bj TC [3] (100G-KR4/CR4/KP4)

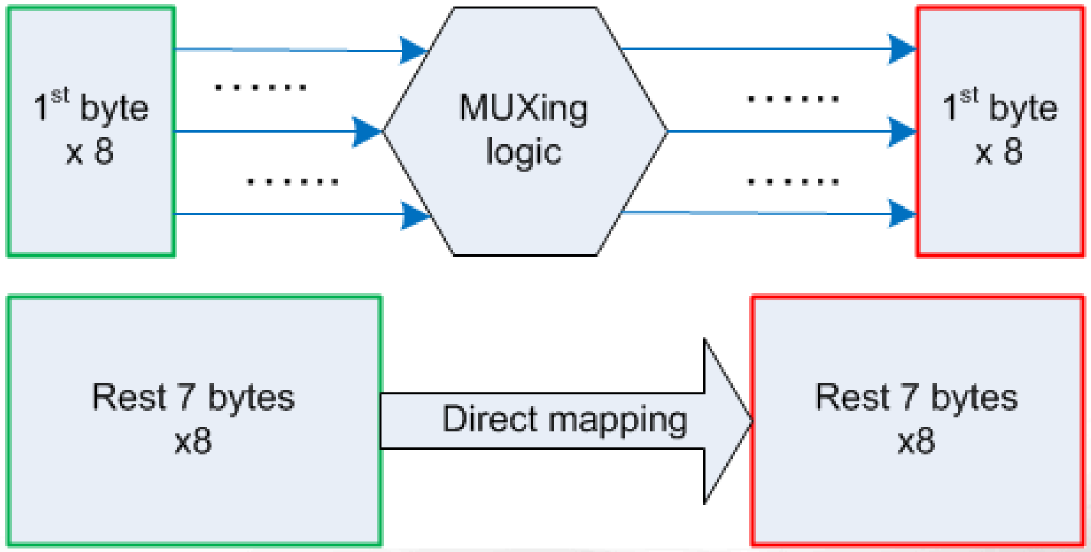
- **Same goal to reduce processing latency**
- However, due to irregular arrangement of first bytes, the rest 7 bytes of top three 64-b blocks do not have fixed mapping locations in transcoding. Thus the MUXing logic is still needed for most of those data.
- On the other hand, **this method can’t be simply extended to 512b/513b case** since we need 8 bits to label 8 64-b blocks while we can only squeezing 4 bits from one control block.



[3] http://www.ieee802.org/3/100GNGOPTX/public/mar12/interim/cideciyan_01_0312_NG100GOPTX.pdf

Transcoding of Data Blocks

- When dealing with 8 65-b pure data blocks, to ease the implementation, we should first transmit the first byte of each 64-b block as we did for those cases with one or more control blocks.
- In this way, there are fixed mapping locations for each 64-bit block except the first byte.



Analyses

- Since the muxing logic is reduced by approximately 8 times in the proposed transcoding scheme, a big (ratio) reduction on power and complexity is feasible.

- For 40G data rate, a parallel level of 64 or 128 (i.e., 128bits/cyc) can be assumed considering moderate clock speed. Hence, the trans-decoding latency can be reduced by approximately 10ns. Although being small to overall latency budget of 40GBase-T, it should be significant for signal processing latency (a part that can be optimized) at PHY layer, e.g., LDPC decoding may take 100ns.

- In brief, the proposed scheme can reduce HW complexity and processing latency. It only requires minor changes to the current spec.

Connecting everything®

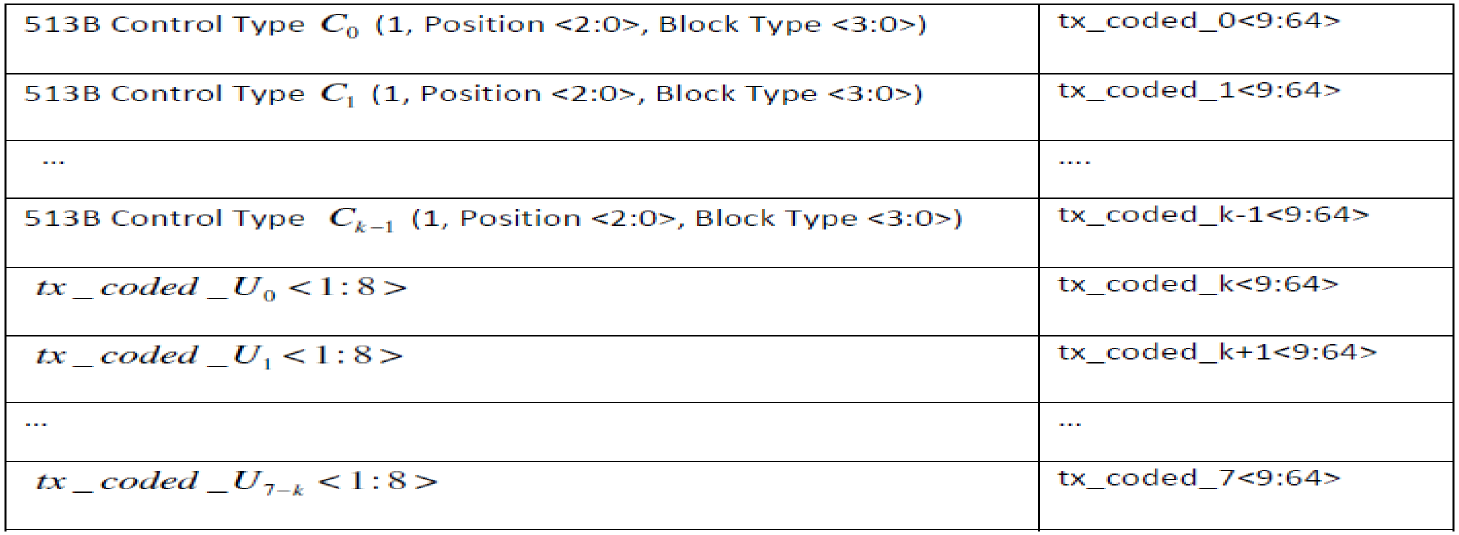
Minor Changes to Spec

- **Change Lines 43-45 on page 98 as follows:**

If for all j=0 to 7, tx_coded_j<0> = 0, tx_xcoded<512:0> shall be constructed as follows:

a) tx_xcoded<0>=1 ,
b) tx_xcoded<(8j+8):(8j+1)>=tx_coded_j<8:1> for j=0 to 7,
c) tx_xcoded<(64+56j+56):(64+56j+1)>=tx_coded_j<64:9> for j=0 to 7.

- **Change Table 113-2 as follows:**



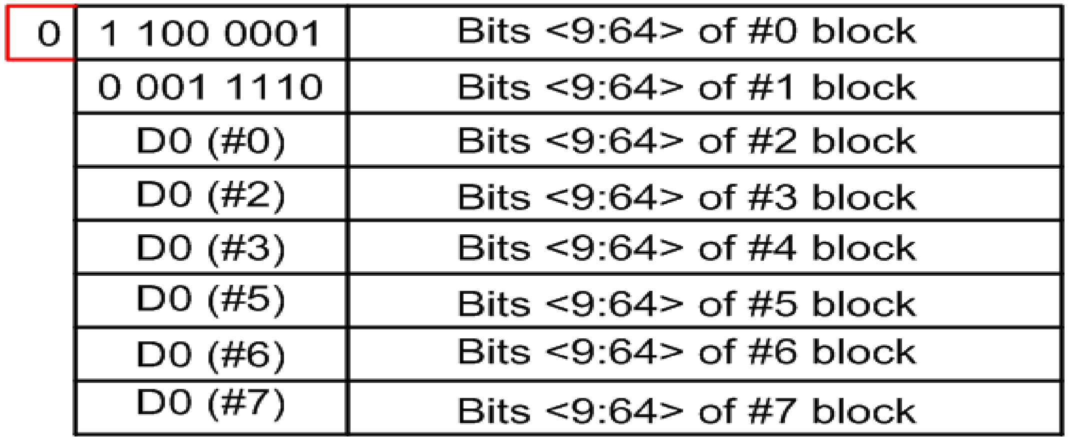
513B Control Type C_0 (1, Position <2:0>, Block Type <3:0>)	tx_coded_0<9:64>
513B Control Type C_1 (1, Position <2:0>, Block Type <3:0>)	tx_coded_1<9:64>
...	....
513B Control Type C_{k-1} (1, Position <2:0>, Block Type <3:0>)	tx_coded_k-1<9:64>
$tx\_coded\_U_0 <1:8>$	tx_coded_k<9:64>
$tx\_coded\_U_1 <1:8>$	tx_coded_k+1<9:64>
...	...
$tx\_coded\_U_{7-k} <1:8>$	tx_coded_7<9:64>

- **Add following texts after line 20 on page 99:**

Similar to the above case with pure data blocks, the first bytes of 8 blocks are mapped to *tx_xcoded<64:1>*, **e.g., the first byte of the first block are mapped to** *tx_xcoded<8:1>*, **and** $tx\_coded\_U_{7-k} <1:8> => tx\_xcoded <57:64>$. **The rest data are mapped to** *tx_xcoded<512:65>*.

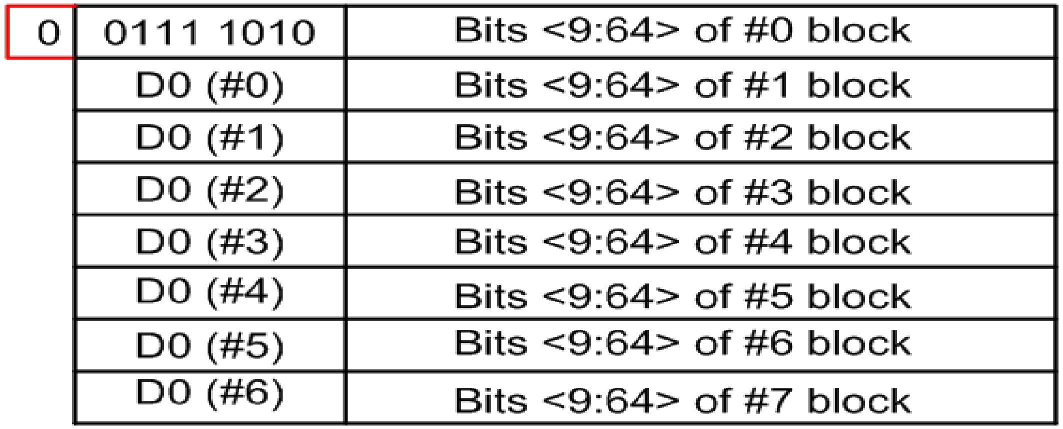
Option-II: Minor Changes (II)

- Change table for Example 1 on page 101 as follows:



0	1 100 0001	Bits <9:64> of #0 block
	0 001 1110	Bits <9:64> of #1 block
	D0 (#0)	Bits <9:64> of #2 block
	D0 (#2)	Bits <9:64> of #3 block
	D0 (#3)	Bits <9:64> of #4 block
	D0 (#5)	Bits <9:64> of #5 block
	D0 (#6)	Bits <9:64> of #6 block
	D0 (#7)	Bits <9:64> of #7 block

- Change table for Example 2 on page 101 as follows:



0	0111 1010	Bits <9:64> of #0 block
	D0 (#0)	Bits <9:64> of #1 block
	D0 (#1)	Bits <9:64> of #2 block
	D0 (#2)	Bits <9:64> of #3 block
	D0 (#3)	Bits <9:64> of #4 block
	D0 (#4)	Bits <9:64> of #5 block
	D0 (#5)	Bits <9:64> of #6 block
	D0 (#6)	Bits <9:64> of #7 block