Short Tutorial on Transcoder Options

William Lo, Axonne Inc.

July 12, 2018

The Encoders Under Consideration

- 64/65 from 10GBASE-T

- 512/513 from 25GBASE-T

- $8n/(8n+1)$ from 1000BASE-T1
  - n=10 for 1000BASE-T1
  - n=16 proposed for MGBASE-T1

64/65 Encoder

1 bit Control/Data Block

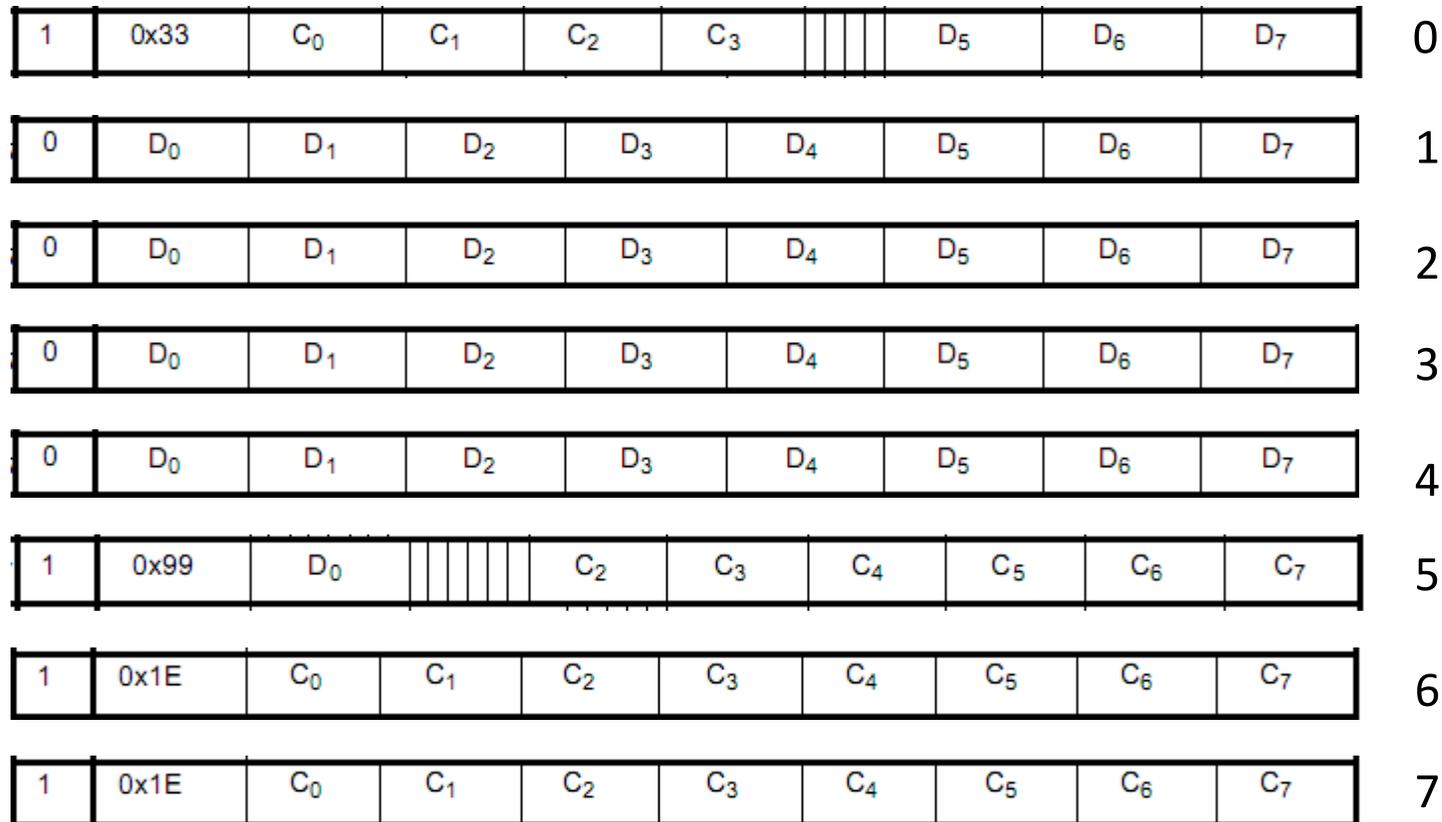**64 bit payload**

0 = Data Only Block

1 = At least one control code

1) Decode 2 x 36 bits on XGMII

2) Determine which of 15 patterns

3) Map XGMII Cx, to payload version

4) Output the mapped Cx, Ox, Dx according to 1 of 15 patterns

Fig 55-9

Input Data	data ctrl header	Block Payload							
Bit Position: 0	1								64
Data Block Format:									
$D_0 D_1 D_2 D_3/D_4 D_5 D_6 D_7$	0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
Control Block Formats:		Block							
$C_0 C_1 C_2 C_3/C_4 C_5 C_6 C_7$	1	0x1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
$C_0 C_1 C_2 C_3/O_4 D_5 D_6 D_7$	1	0x2D	C_0	C_1	C_2	C_3	O_4	D_5	D_6	D_7
$C_0 C_1 C_2 C_3/S_4 D_5 D_6 D_7$	1	0x33	C_0	C_1	C_2	C_3		D_5	D_6	D_7
$O_0 D_1 D_2 D_3/S_4 D_5 D_6 D_7$	1	0x66	D_1	D_2	D_3	O_0		D_5	D_6	D_7
$O_0 D_1 D_2 D_3/O_4 D_5 D_6 D_7$	1	0x55	D_1	D_2	D_3	O_0	O_4	D_5	D_6	D_7
$S_0 D_1 D_2 D_3/D_4 D_5 D_6 D_7$	1	0x78	D_1	D_2	D_3	D_4	D_5	D_6	D_7	
$O_0 D_1 D_2 D_3/C_4 C_5 C_6 C_7$	1	0x4B	D_1	D_2	D_3	O_0	C_4	C_5	C_6	C_7
$T_0 C_1 C_2 C_3/C_4 C_5 C_6 C_7$	1	0x87		C_1	C_2	C_3	C_4	C_5	C_6	C_7
$D_0 T_1 C_2 C_3/C_4 C_5 C_6 C_7$	1	0x99	D_0		C_2	C_3	C_4	C_5	C_6	C_7
$D_0 D_1 T_2 C_3/C_4 C_5 C_6 C_7$	1	0xAA	D_0	D_1		C_3	C_4	C_5	C_6	C_7
$D_0 D_1 D_2 T_3/C_4 C_5 C_6 C_7$	1	0xB4	D_0	D_1	D_2		C_4	C_5	C_6	C_7
$D_0 D_1 D_2 D_3/T_4 C_5 C_6 C_7$	1	0xCC	D_0	D_1	D_2	D_3		C_5	C_6	C_7
$D_0 D_1 D_2 D_3/D_4 T_5 C_6 C_7$	1	0xD2	D_0	D_1	D_2	D_3	D_4		C_6	C_7
$D_0 D_1 D_2 D_3/D_4 D_5 T_6 C_7$	1	0xE1	D_0	D_1	D_2	D_3	D_4	D_5		C_7
$D_0 D_1 D_2 D_3/D_4 D_5 D_6 T_7$	1	0xFF	D_0	D_1	D_2	D_3	D_4	D_5	D_6	

AXONNE

Clause 133.3.2.2.15   512/513 Transcoder (step 1)

1) Convert 16 XGMII into 8 64/65 blocks

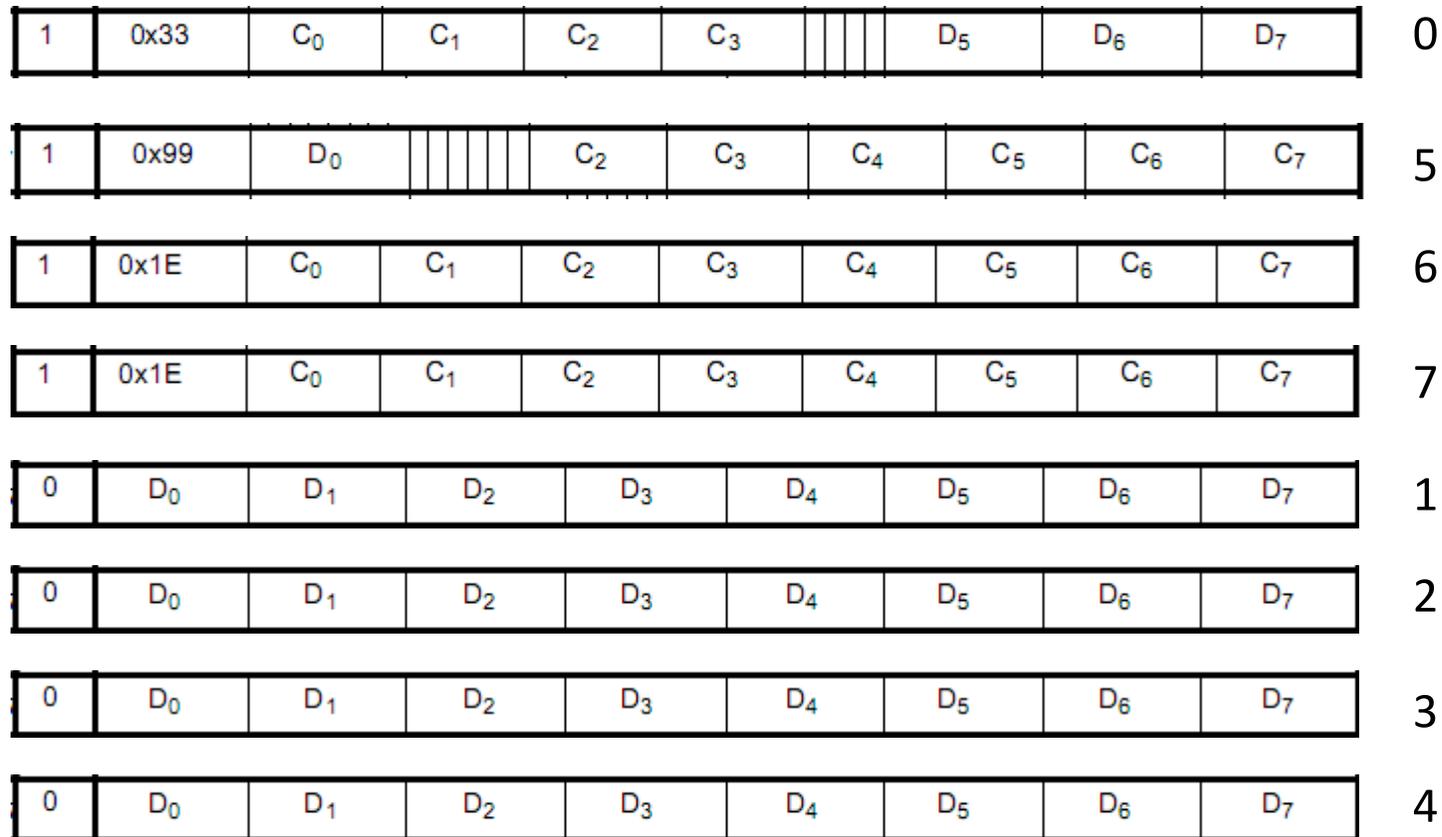| 1	0x33	C_0	C_1	C_2	C_3		D_5	D_6	D_7	0
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		1
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		2
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		3
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		4
1	0x99	D_0		C_2	C_3	C_4	C_5	C_6	C_7	5
1	0x1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	6
1	0x1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	7

512/513 Transcoder (step 2)

2) Reorder 64/65 blocks so that control blocks appears first

 - Intuitive for humans to do

  - Lots of gates for circuit to do

1	0x33	C_0	C_1	C_2	C_3	‖‖‖	D_5	D_6	D_7	0
1	0x99	D_0	‖‖‖‖	C_2	C_3	C_4	C_5	C_6	C_7	5
1	0x1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	6
1	0x1E	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	7
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		1
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		2
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		3
0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7		4

AXONNE

512/513 Transcoder (step 3)

0 = At least one 64/65 control block

1 = All 64/65 data block

Remap 8 bits according to 113.3.2.2.15

Remove control/data bit from all 64/65 blocks
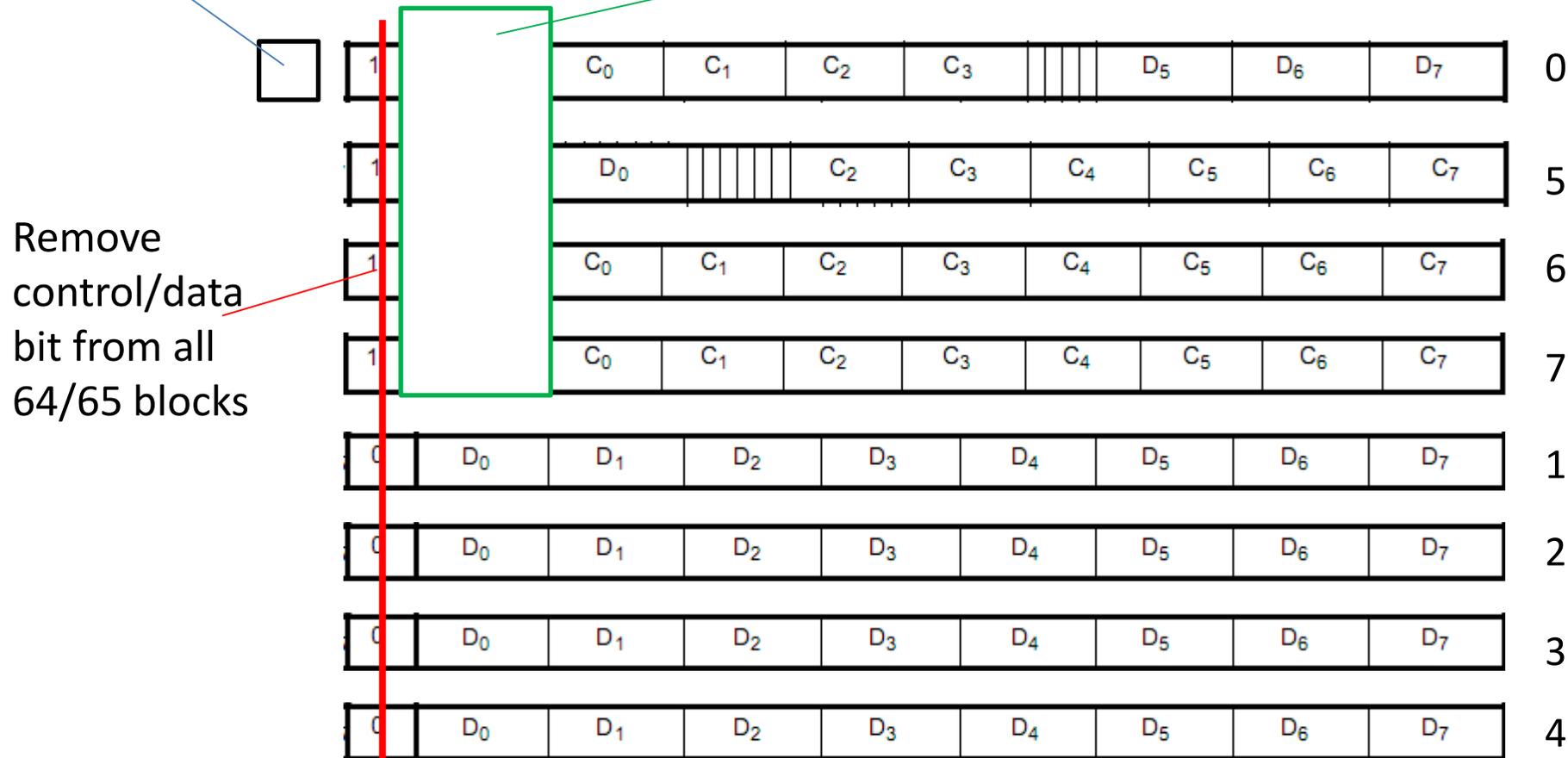
Clause 97.3.2.2.5   8n/8n+1 coding

- Using n = 8 as example

▶ **Instead of using fixed block types, use pointers instead**

▶ **No change if all bytes are data bytes – same as before**

0	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7
	0 ••• 7	0 ••• 7	0 ••• 7	0 ••• 7	0 ••• 7	0 ••• 7	0 ••• 7	0 ••• 7
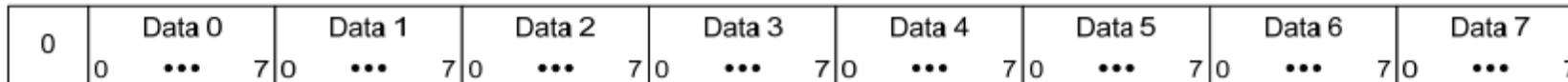
▶ **If byte is control byte use 5 bit pointer + 3 bit control code**

▶ **If byte is data use 8 bit data**

▶ **Bit 0 to 3 of pointer points to next byte that is a control symbol**

▶ **Bit 4 of pointer indicates whether the next control symbol is the final control symbol of the block**

- ▪ 0 = final one, 1 = more control symbols

AXONNE
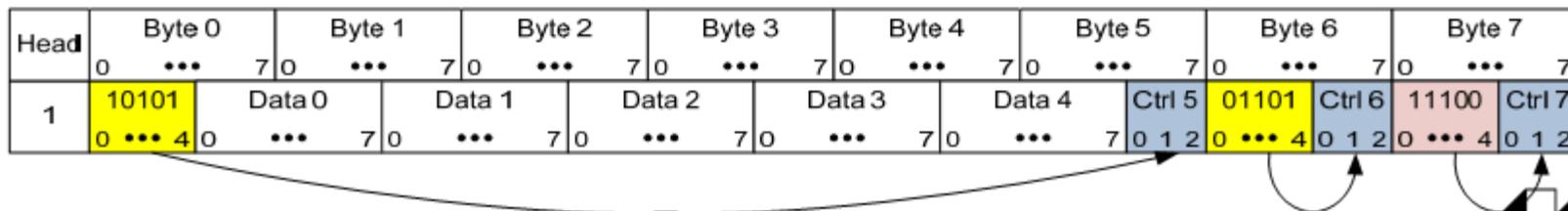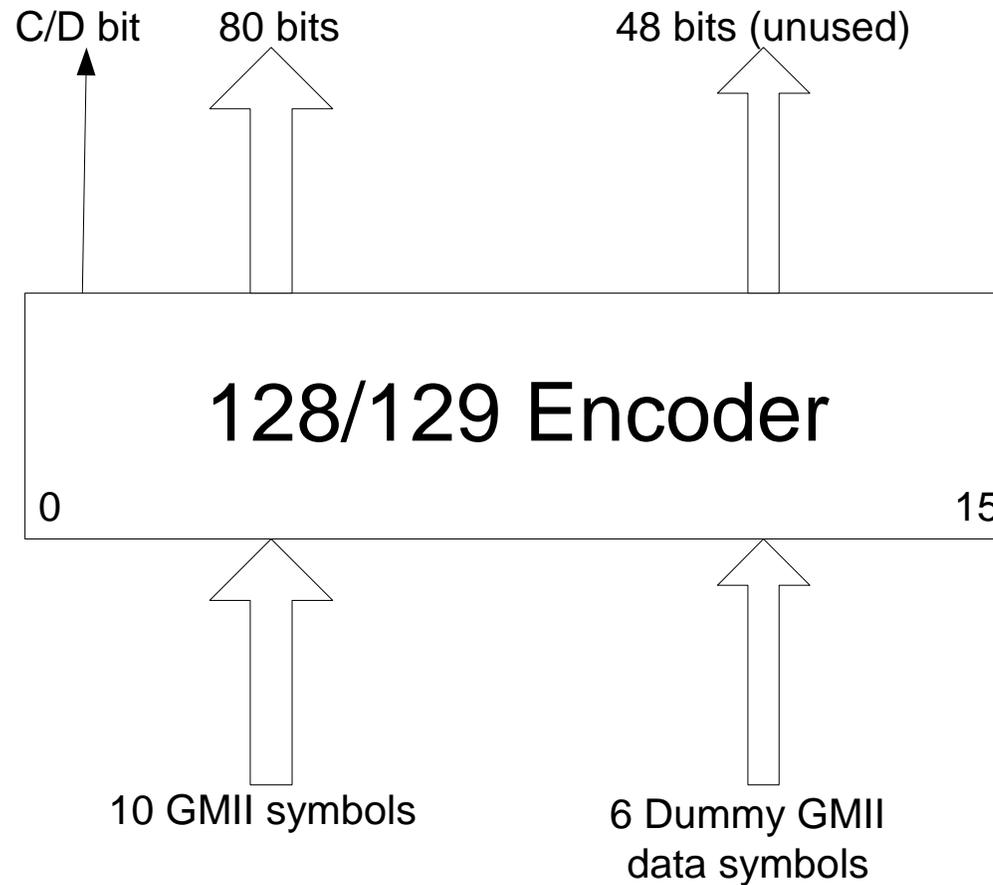
8n/8n+1 coding – Control block examples
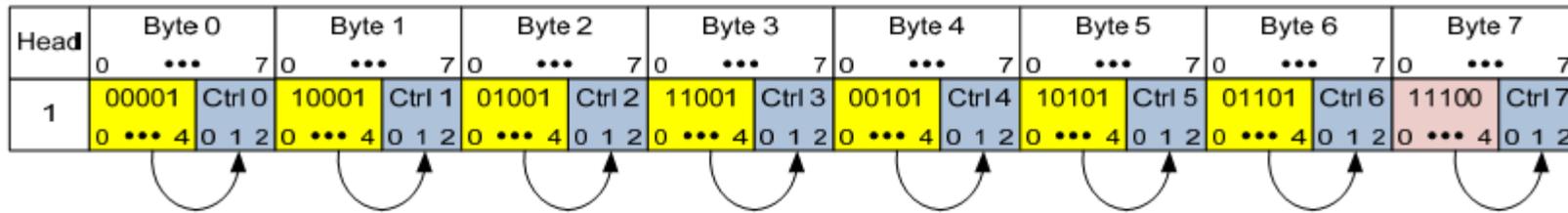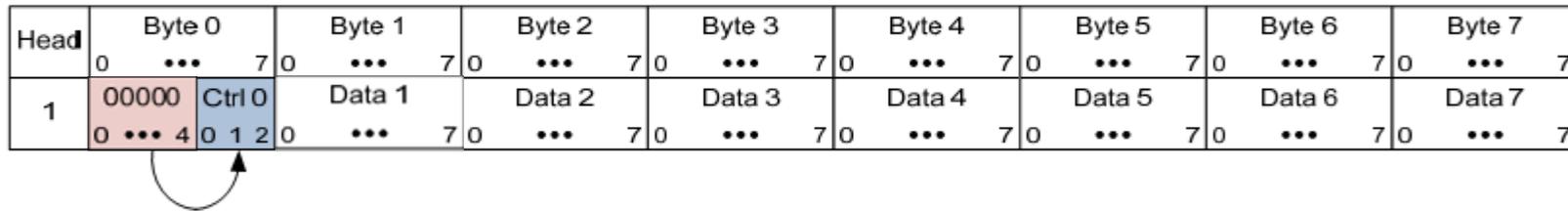
128/129 Used as 80/81 for 1000BASE-T1

THANK YOU