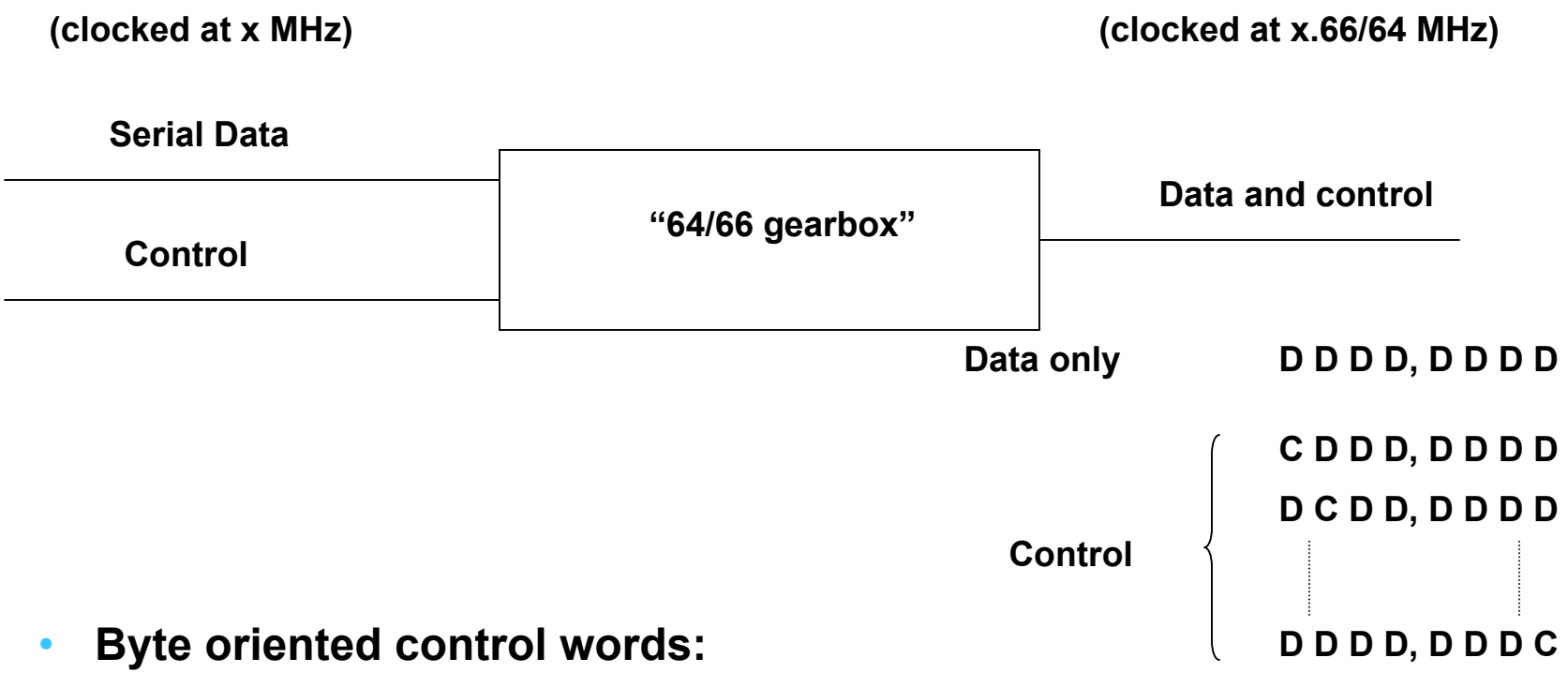
Encapsulation proposal

Hugh Barrass

The story so far…

- ITU-T proposed (and adopted) HDLC framing for packet transport
  - See as the simplest and easiest method
- Very strong negative reaction from “Ethernet antibodies”
  - HDLC has been rejected multiple previous times
  - Concerns regarding variable payload rates and undetected error rate
- Detailed analysis performed
  - Undetected error rate not proven
  - Alternatives viewed and “better” options considered
- Draft 1.0 captures the indecision
  - 61.2.3 (editor’s note): “it has yet to be decided”
  - This issue must be closed ASAP!

Encapsulation 64b/66b



- **Byte oriented control words:**
  - 2^{64} data only codewords
  - 2^{59} codewords for each control code
  - As used for 10Gig Ethernet
- **Overhead = 2/64 (~3%) + at least 1 byte per frame (SOP)**

64b/66b in 802.3ae

- **64b/66b offered the best compromise between robustness, overhead and complexity**
  - Detailed presentations available
  - Published standard includes precise implementation
- **Some differences between requirements for .3ae and EFMCu**
  - A slightly modified version proposed
- **(no) Preservation of Ethernet IPG**
  - .3ae keeps 160 bit gap between last bit & first bit (preamble + IPG)
  - EFMCu has preference to minimize IPG (& eliminate preamble)
- **Scrambling**
  - .3ae includes scrambler in PCS, EFMCu has scrambler in PMA

Primary difference – coding table

Type	sync	Byte 0	1	2	3	4	5	6	7
Data only	01	D0	D1	D2	D3	D4	D5	D6	D7
Idle only	10	0x1e	x1	x2	x3	x4	x5	x6	x7
Start	10	0x33	sPos (1-7)	D2	D3	D4	D5	D6	D7
Start(0)	10	0x78	D1	D2	D3	D4	D5	D6	D7
Terminate	10	0x87	tPos (0-6)	D0	D1	D2	D3	D4	D5
Terminate(7)	10	0x99	D6	D0	D1	D2	D3	D4	D5
T-S	10	0xaa	sPos(1-7) tPos(0-6)	D0/D2	D1/D3	D2/D4	D3/D5	D4/D6	D5/D7

- Pos indicates the position of the Start or Terminate flag

Data before start or after terminate is don't care

Examples(1) – what you see…

- **Mid frame or between frames – it's easy!**

64 data bits become 66 bit codeword

Byte stream	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
SOP/EOP																

First block

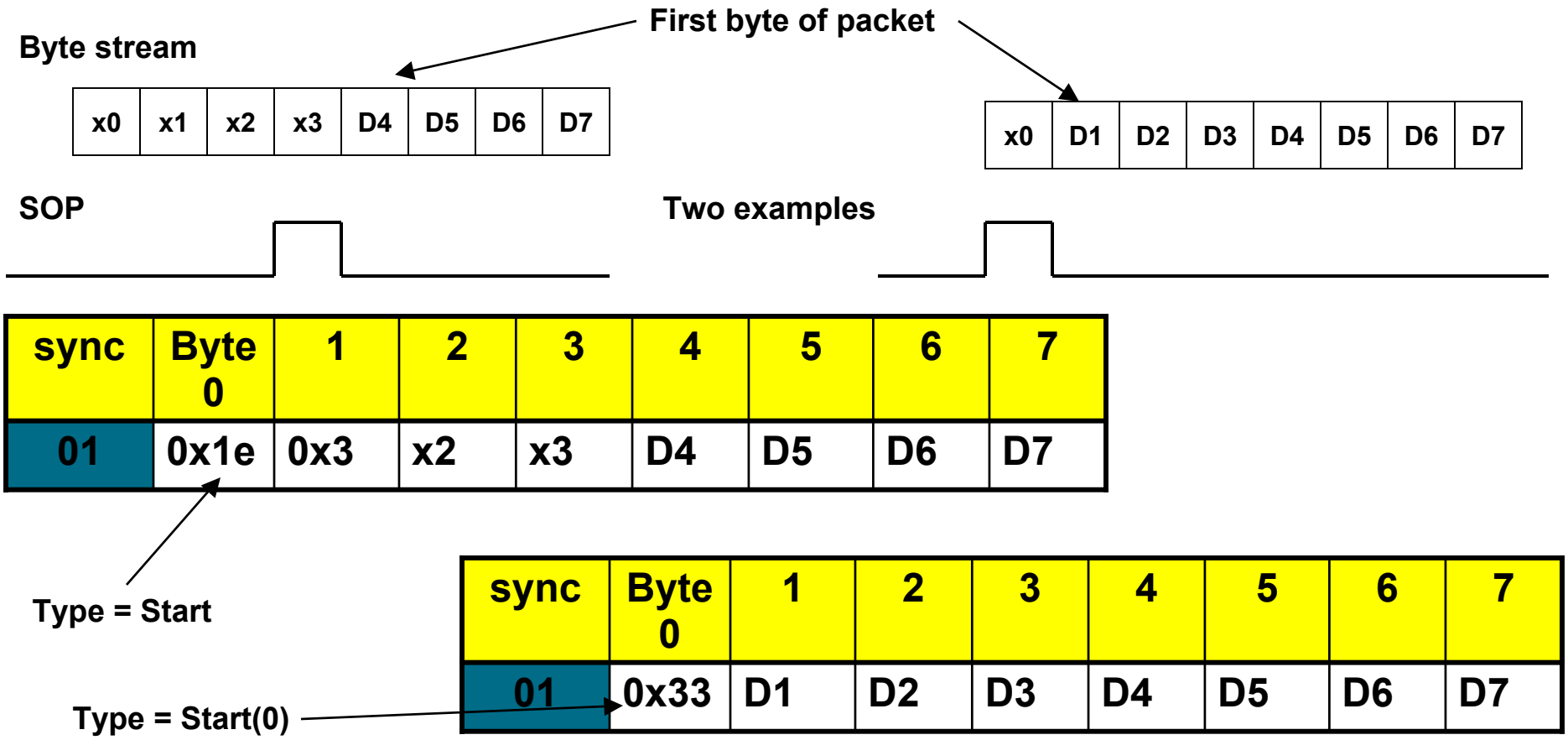
sync	Byte 0	1	2	3	4	5	6	7
01	D0	D1	D2	D3	D4	D5	D6	D7

Second block

sync	Byte 0	1	2	3	4	5	6	7
01	D8	D9	D10	D11	D12	D13	D14	D15

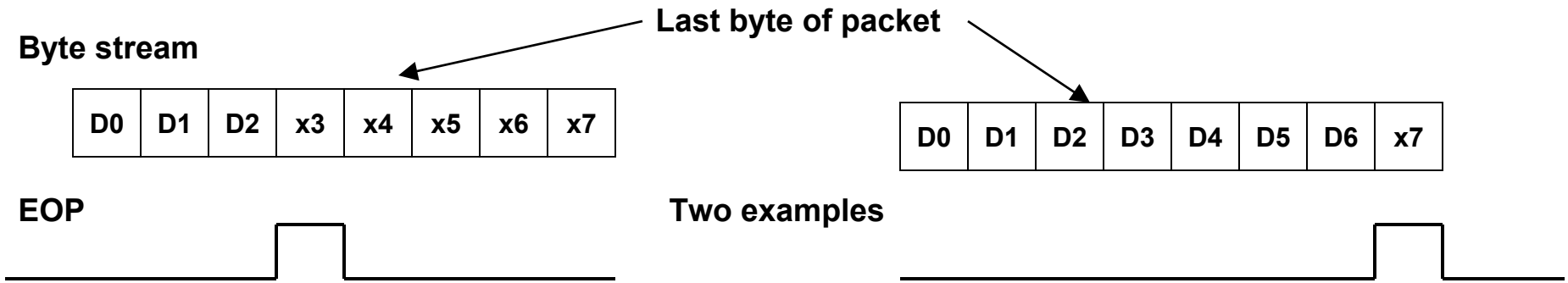
Examples(2) – Start of packets

- Simple start scenarios – more than 7 byte IPG



Examples(3) – End of packets

- Simple end scenarios – more than 7 byte IPG



sync	Byte 0	1	2	3	4	5	6	7
01	0x78	0x3	D0	D1	D2	D3	D4	D5

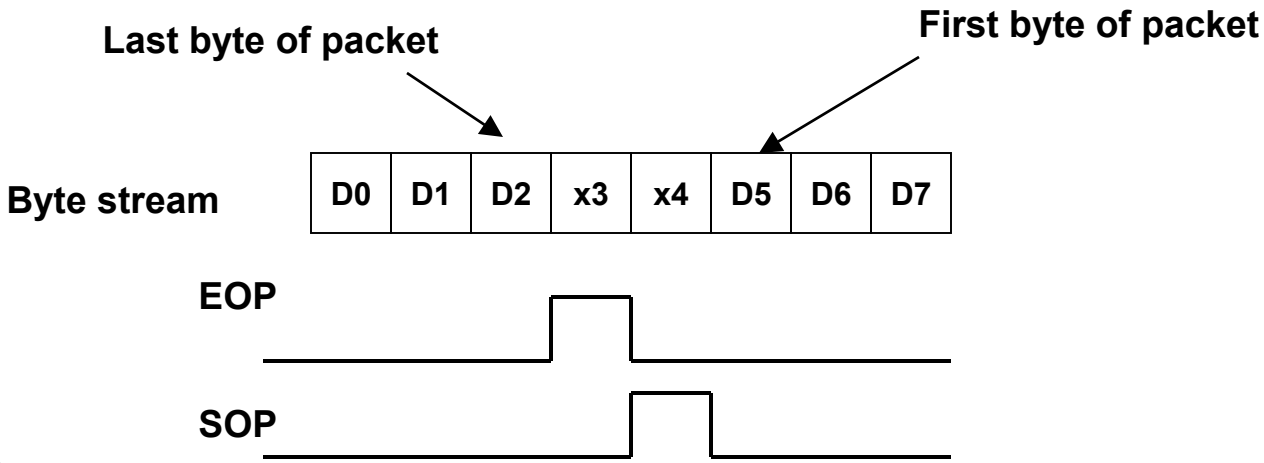
Type = Terminate

sync	Byte 0	1	2	3	4	5	6	7
01	0x87	D6	D0	D1	D2	D3	D4	D5

Type = Terminate(7)

Examples(4) – Short IPG

- Start and terminate in one code block

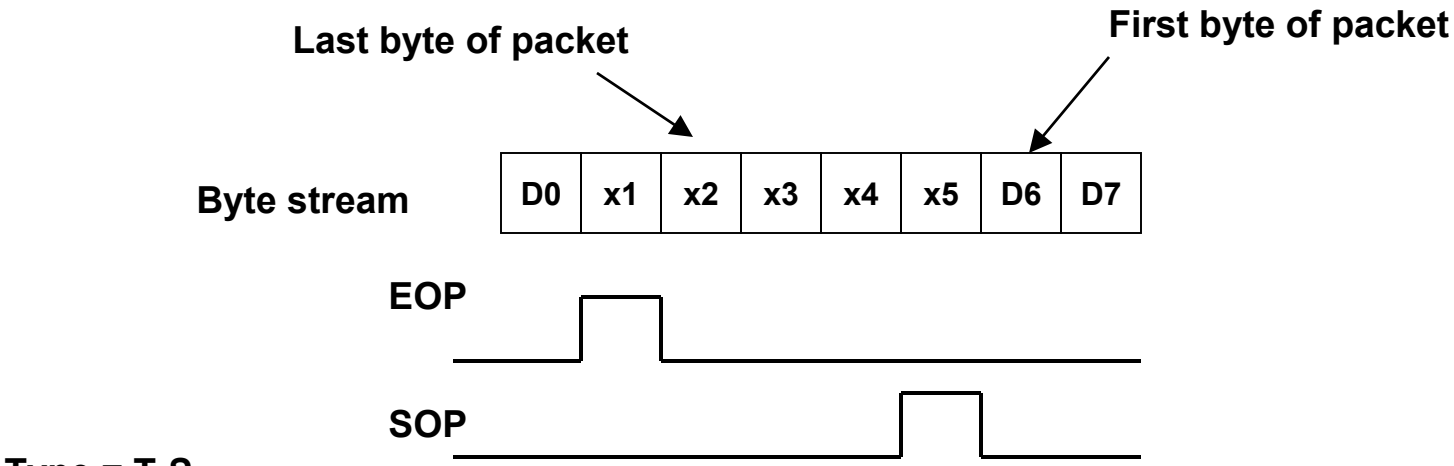


Type = T-S

sync	Byte 0	1	2	3	4	5	6	7
01	0x99	0x43	D0	D1	D2	D5	D6	D7

Examples(5) – Short IPG - again

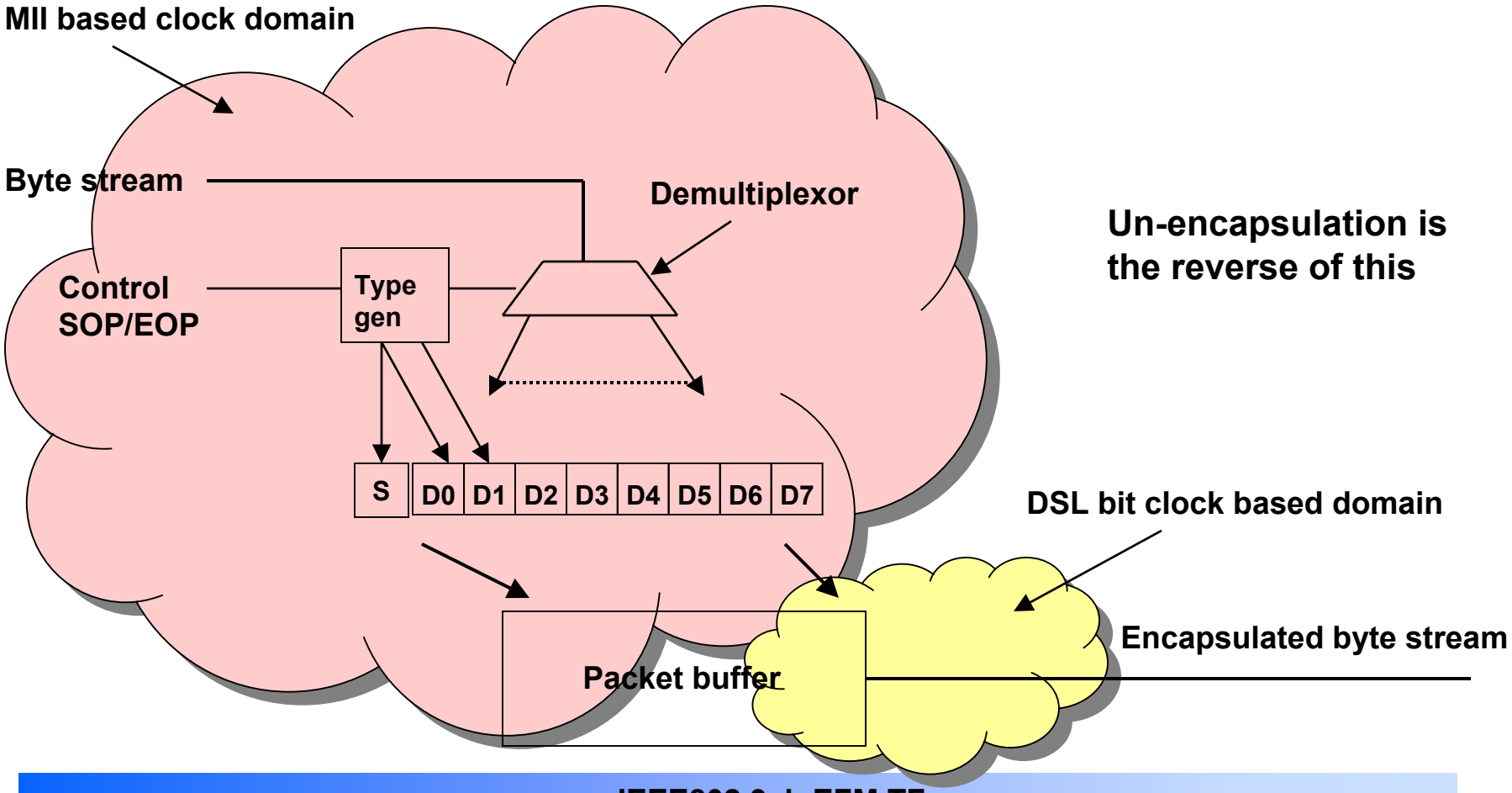
- Different start and terminate in one code block



sync	Byte 0	1	2	3	4	5	6	7
01	0x99	0x52	D0	x1	x2	x5	D6	D7

Implementation example

- It looks complex on paper – but it's easy compared to DSL!

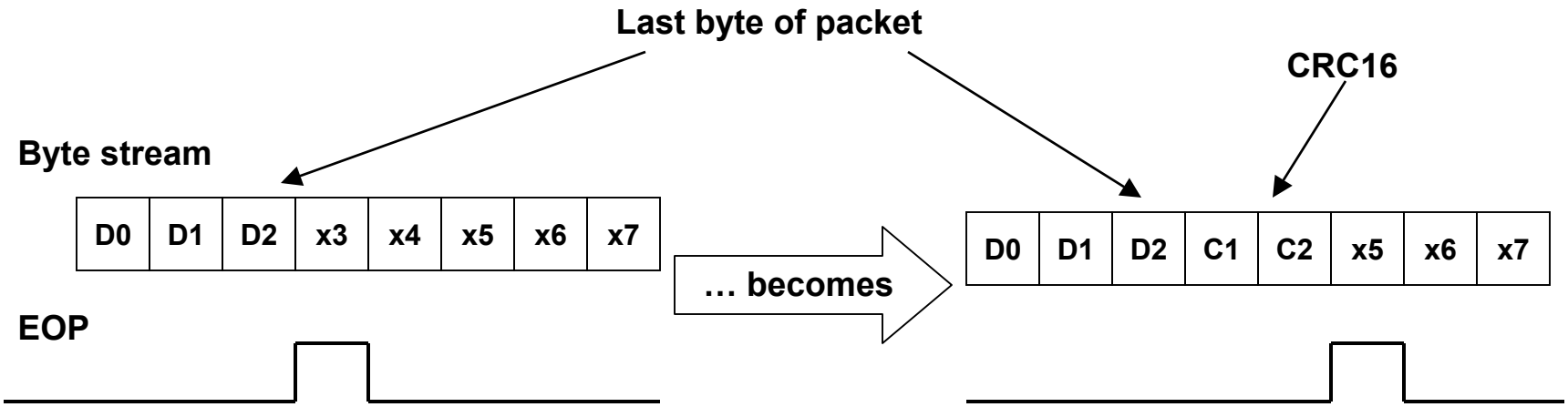


More problems!!!

- **Encapsulation CRC**
  - Barry's analysis (omahony_1_0502.pdf) of False Packet Acceptance
  - Needs 16bit CRC in encapsulation
- **Loop Aggregation header**
  - Extra 3 bytes to add to frame
  - Only for aggregated link – no header added for single
  - Needs more types defined

End of packets with CRC

- **Add 2 byte CRC - after last frame byte**
  - **Simple process, it's always there (so nothing else changes)**



Type = Terminate

sync	Byte 0	1	2	3	4	5	6	7
01	0x78	0x5	D0	D1	D2	C1	C2	x5

Bigger coding table – 3 new codes

Type	sync	0	1	2	3	4	5	6	7
Data only	01	D0	D1	D2	D3	D4	D5	D6	D7
Idle only	10	0x1e	x1	x2	x3	x4	x5	x6	x7
Start	10	0x33	sPos (1-7)	D2	D3	D4	D5	D6	D7
Start(0)	10	0x78	D1	D2	D3	D4	D5	D6	D7
Terminate	10	0x87	tPos (0-6)	D0	D1	D2	D3	D4	D5
Terminate(7)	10	0x99	D6	D0	D1	D2	D3	D4	D5
T-S	10	0xaa	sPos(1-7) tPos(0-6)	D0/D2	D1/D3	D2/D4	D3/D5	D4/D6	D5/D7
S-LAH	10	0xb4	sPos (1-7)	D2	D3	D4	D5	D6	D7
S(0)-LAH	10	0xcc	D1	D2	D3	D4	D5	D6	D7
TS-LAH	10	0xd2	sPos(1-7) tPos(0-6)	D0/D2	D1/D3	D2/D4	D3/D5	D4/D6	D5/D7

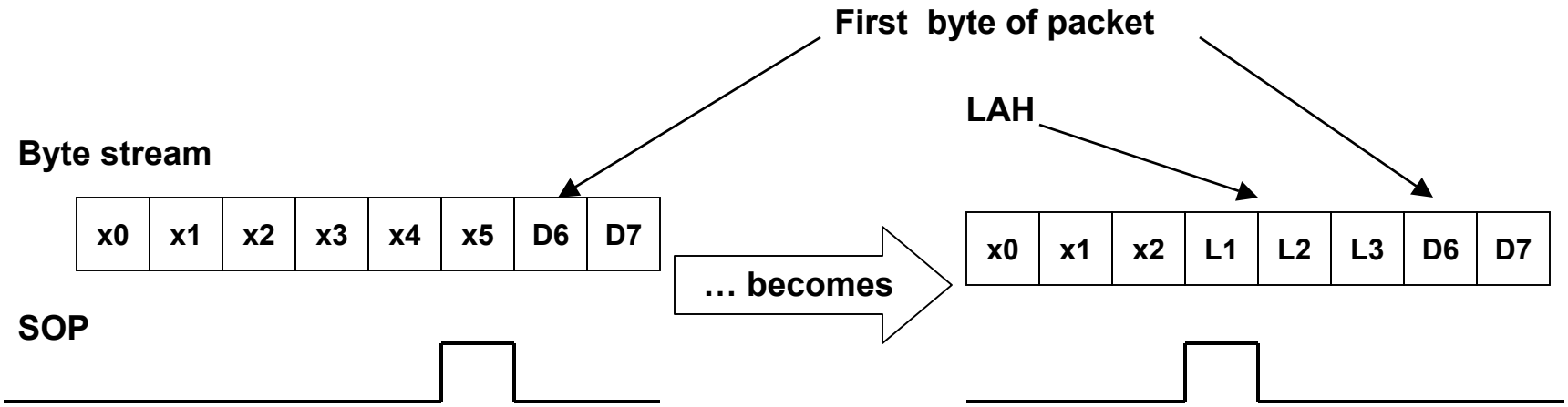
- S-LAH, S(0)-LAH and TS-LAH are identical to Start, Start(0) and T-S

Except that the three byte LAH precedes the first frame byte

Start of packet with LAH

- **Add 3 byte LAH – before first frame byte**

  Simple process, (if) it's always there (so nothing else changes)

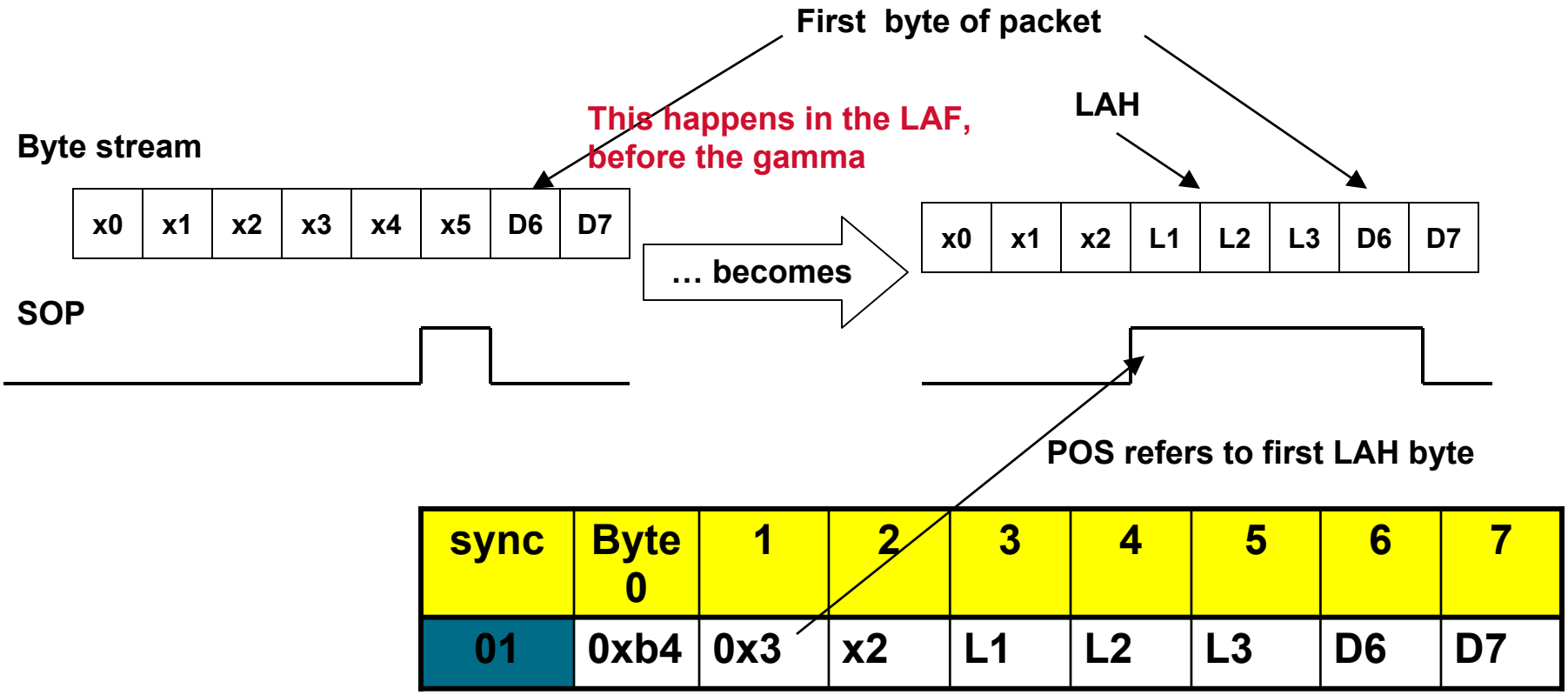


Type = S-LAH

sync	Byte 0	1	2	3	4	5	6	7
01	0xb4	0x3	x2	L1	L2	L3	D6	D7

But how do we know it's LAH?

- Need to identify LAH vs non-LAH across the gamma
  - Separate proposal – use a longer SOP



… so there we have it!

- **Encapsulate with 64b/66b**
  - Similar to 802.3ae
  - Define new types and new multiplexing
  - No need for scrambler
- **Add CRC16 protection**
  - > 1 / 10 billion years undetected frame error rate
- **Support Loop Aggregation Header**
  - Extra 3 bytes when required (0 when not)
- **Complete proposal – soup-to-nuts**
  - Fits between alpha/beta and gamma