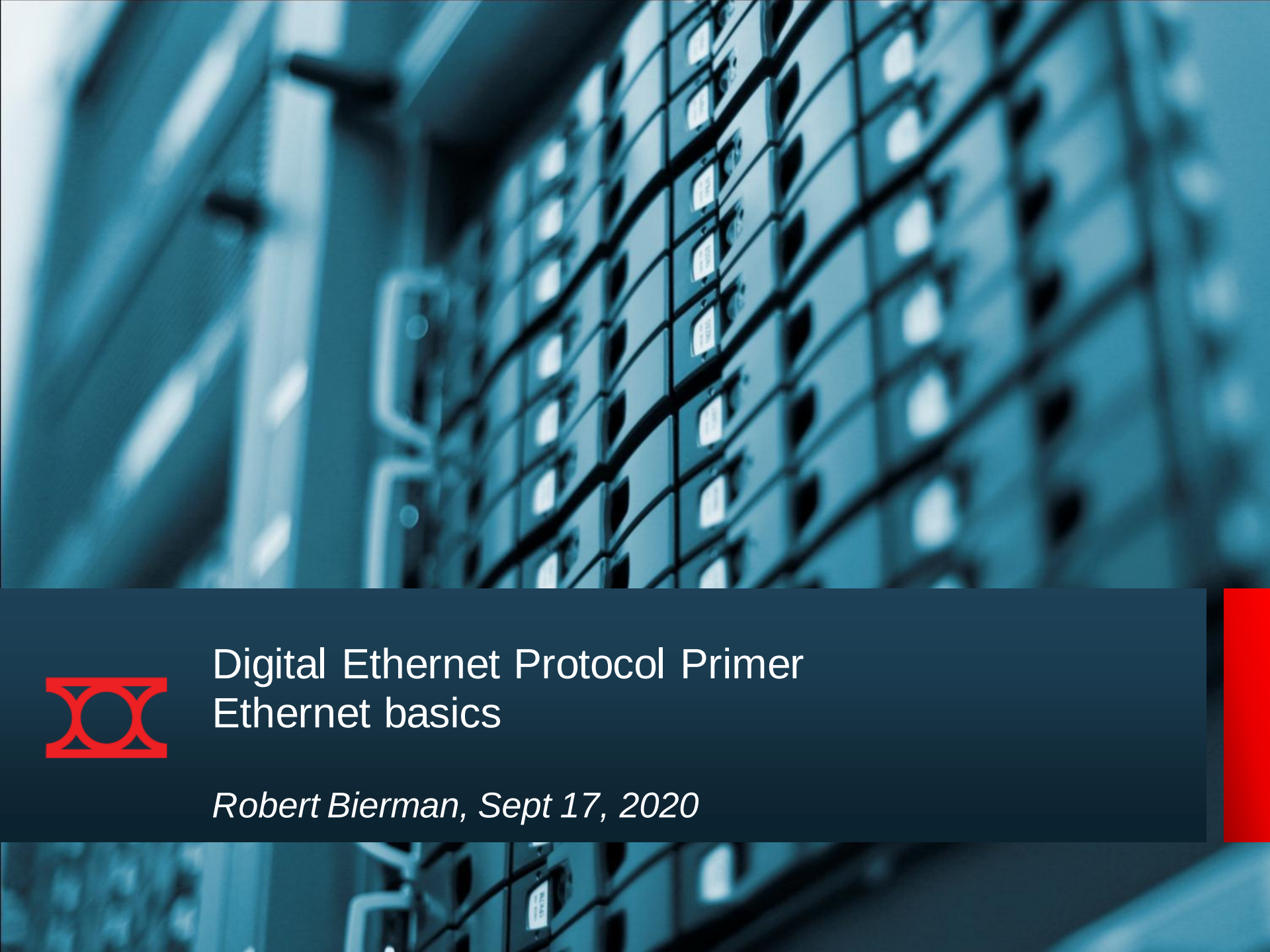
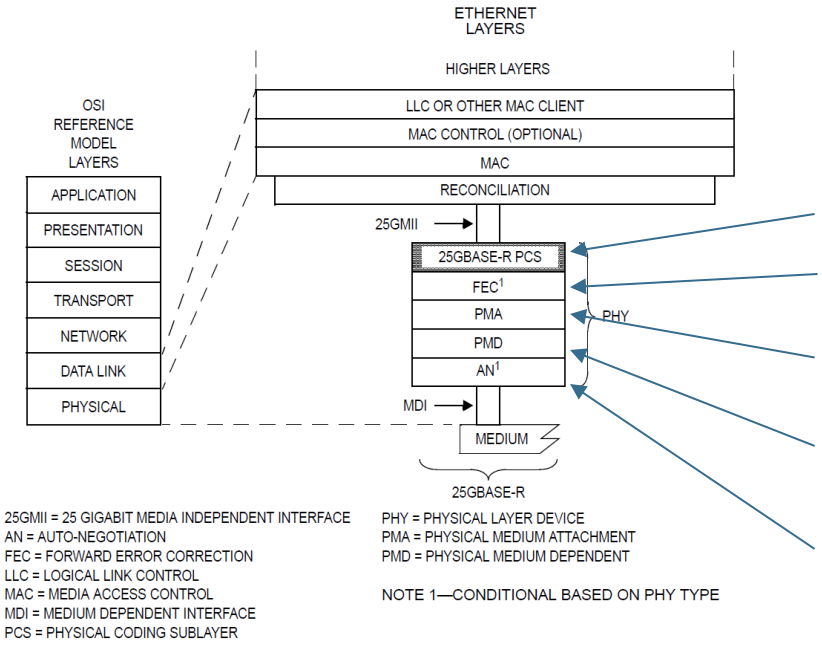


Digital Ethernet Protocol Primer
Ethernet basics

Robert Bierman, Sept 17, 2020

Introduction to Ethernet

- We are mostly interested in the PCS and FEC layers in the “Protocol stack”
- Data moves down the stack, from Switch out to Driver and back



Ethernet encoding

FEC encoding

PMA layer preps data for transmission

e.g. serialize, PAM4 encoding

PMD includes Driver + Receiver

AN = auto-negotiation

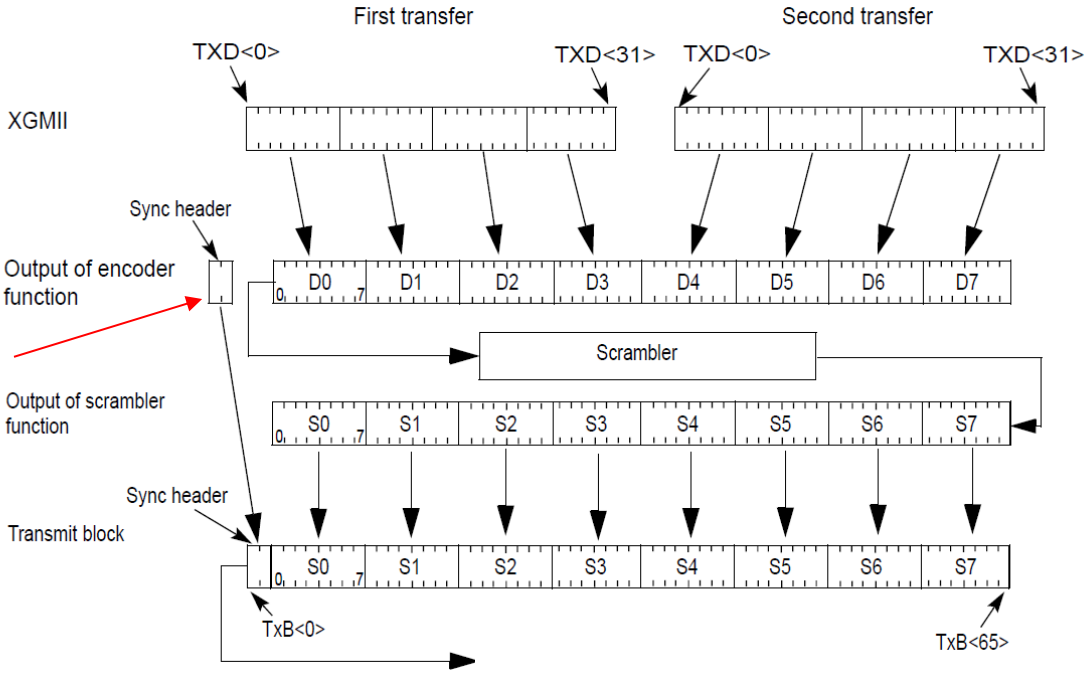
e.g. Vega, Syrma

64b/66b encoding

- Most 10GE data (and higher) uses a 64b/66b encoding scheme (+3.125% overhead)
- 8 Octets (64 bits) are formed into a group, then 2 “sync header” bits added to form into **66-bit Blocks**.
- The Octets are scrambled using a PRBS58 scrambling routine to randomize the transitions. The 2-bit sync header is NOT scrambled.

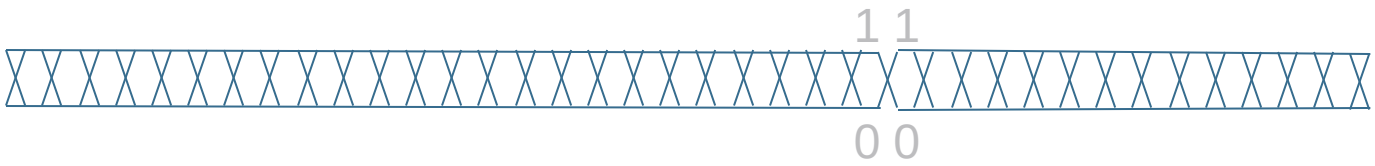
MII = media independent interface
XGMII = the 10G MII

Sync header bypasses the scrambler



Sync header values

- The Sync header is 2 bits long but there are only 2 legal values for it:
- **01**: the 66-bit block carries **data**
- **10**: the 66-bit block carries **control** characters
- Values of 00 and 11 are illegal. If these values are observed e.g. bit error, they will be replaced with an Error /E/ character.
- Because the sync header is not scrambled, it can easily be identified on a scope



- Q. What purpose does the Sync Header serve?
- Ans. Its primary purpose is to enable data alignment at the receiver. The sync header is easy to identify in a digital signal since it has only two possible values. The second purpose is to convey the information type in the 66b Block (data or control)

8b10b Encoding below 10G

- 8b/10b encoding is used for rates below 10G
- Includes 100M, 1G, 2.5G (XAUI)
- +25% bandwidth overhead
- With 8 bits, it's possible to define all possible encodings in a table (Table 36-1a)
- Two 10-bit encodings are defined for each Octet, a positive parity and a negative parity
- Protocol alternates between + and – parities on successive symbols

Table 36–1a—Valid data code-groups

Code Group Name	Octet Value	Octet Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D0.0	00	000 00000	100111 0100	011000 1011
D1.0	01	000 00001	011101 0100	100010 1011
D2.0	02	000 00010	101101 0100	010010 1011
D3.0	03	000 00011	110001 1011	110001 0100
D4.0	04	000 00100	110101 0100	001010 1011
D5.0	05	000 00101	101001 1011	101001 0100
D6.0	06	000 00110	011001 1011	011001 0100
D7.0	07	000 00111	111000 1011	000111 0100
D8.0	08	000 01000	111001 0100	000110 1011
D9.0	09	000 01001	100101 1011	100101 0100
D10.0	0A	000 01010	010101 1011	010101 0100
D11.0	0B	000 01011	110100 1011	110100 0100
D12.0	0C	000 01100	001101 1011	001101 0100
D13.0	0D	000 01101	101100 1011	101100 0100

etc

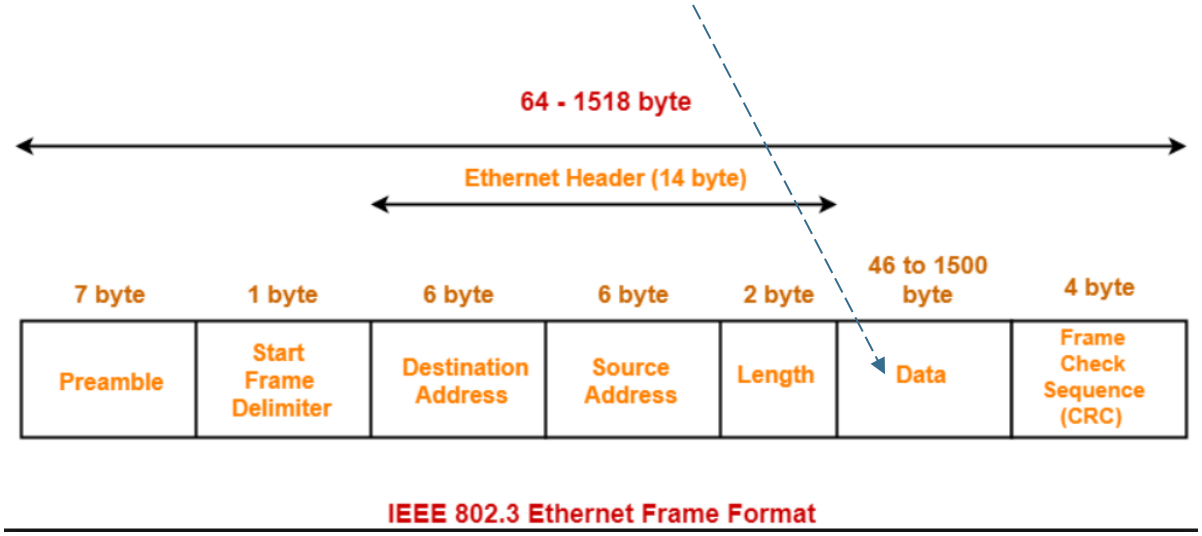
What does the sync header do to data rate?

- The raw data rate for 10GE is 10.0000Gbps +-100ppm
- The raw data rate for 25GE is 25.0000Gbps +- 100ppm
- This is the raw octet data rate. Data rate increases by 66/64 due to sync header

Protocol	Payload rate Gbps	Ratio	Data rate Gbps
25GE	25.0000	66/64	**25.78125**
10GE	10.0000	66/64	**10.3125**
2.5G	2.50000	10/8	**3.125**
1G	1.0000	10/8	**1.25**

Ethernet Frames

- Ethernet transmits data inside Ethernet **Frames**
- Frame contains a Start, Source MAC address, Destination MAC address, a payload (the Data), a CRC
- The data is usually TCP/IP data **packets** but can be other stuff



The maximum Ethernet Frame length is usually 1518 bytes but the protocol allows for **Jumbo frames** with 9000 bytes of data (9018 total frame size).

There is a mistake in the picture above. Can you spot it?

Interpacket Gap

- When there is no Data to send, the MAC sends a filler pattern to fill the void between Ethernet Frames.
- This void is called **"inter-frame gap" or IFG** (sometimes called **"interpacket gap" or IPG**)
- The filler pattern contains Control characters that include Idles (the name is self-explanatory) and some special characters (e.g. Remote Fault, Local Fault)
- IPG can be **inserted or deleted** for rate compensation e.g. ppm differences. Add/delete 4 bytes at a time. Use RATE ADAPT FIFO to manage.

Q. what is the purpose of IPG?

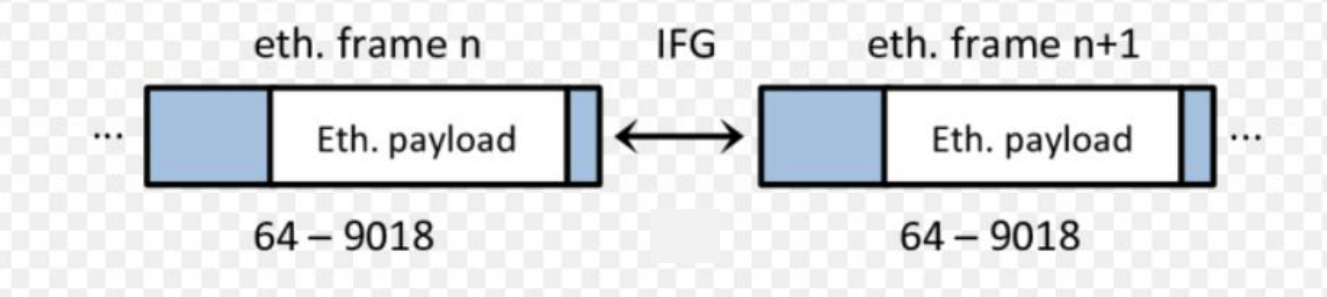
A. It is required to ensure a continuous signal is sent e.g so CDR remains locked.

Q. How long is the IPG? Is there a maximum?

A. It can be infinitely long.

Q. Is there a minimum?

A. Minimum IPG generated by PCS is 12 bytes. However, IPG can be deleted for rate adaptation to an absolute minimum of 8 bytes (?)



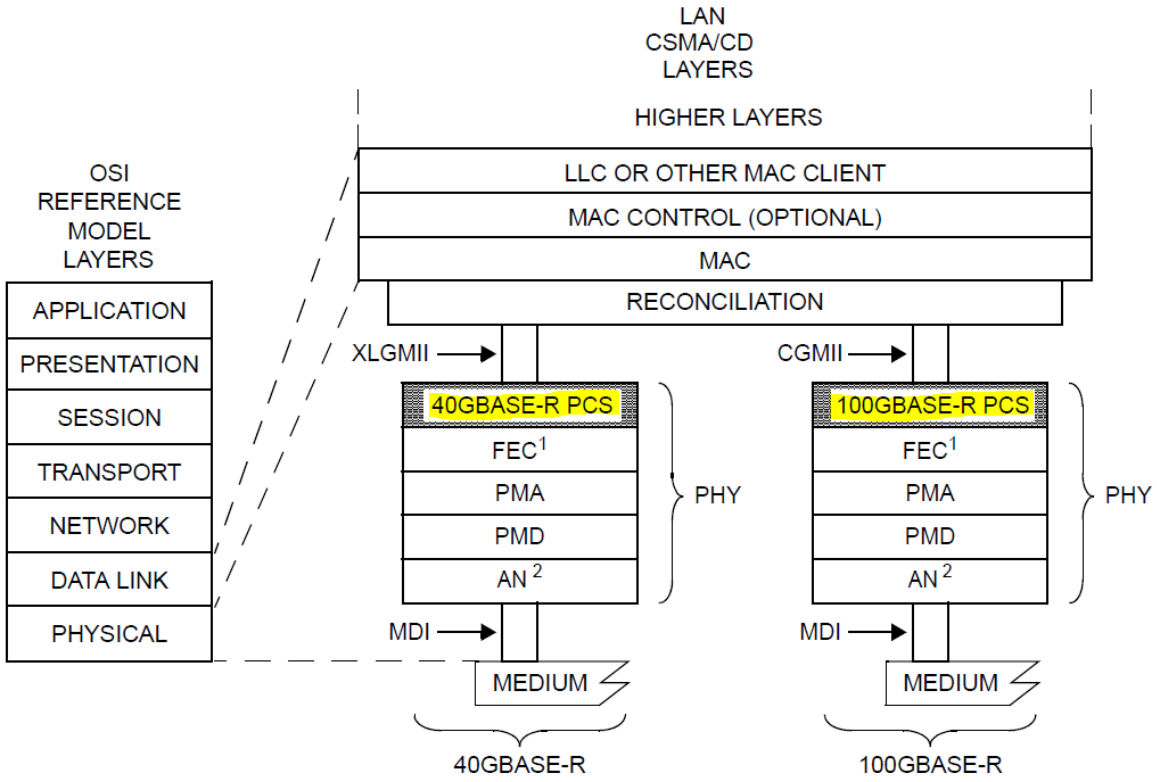
Handling Multiple Lanes

More bandwidth, please!!!

- Max baud rate limits the bandwidth of a single lane protocol
- At some media-independent layer at the MAC, they want a bigger “pipe”
- Easy to define serial 1GE, 10GE, 25GE but you need to take advantage of multiple parallel lanes to get higher bandwidth
- What about PAM4? This gets us to 50GE. But the protocol must support the ability to convert 50GE PAM4 to two NRZ lanes, so handling multiple lanes needs to be built into the protocol.

Multi-Lane Protocols

- First defined multi-lane protocols are 40GE and 100GE (802.3ba)
- Ethernet Frames get distributed across all lanes



Multi-Lane Protocols

Same 64b/66b Block Generation

Distribute blocks into N serial Streams
40GE: N = 4
100GE: N = 20

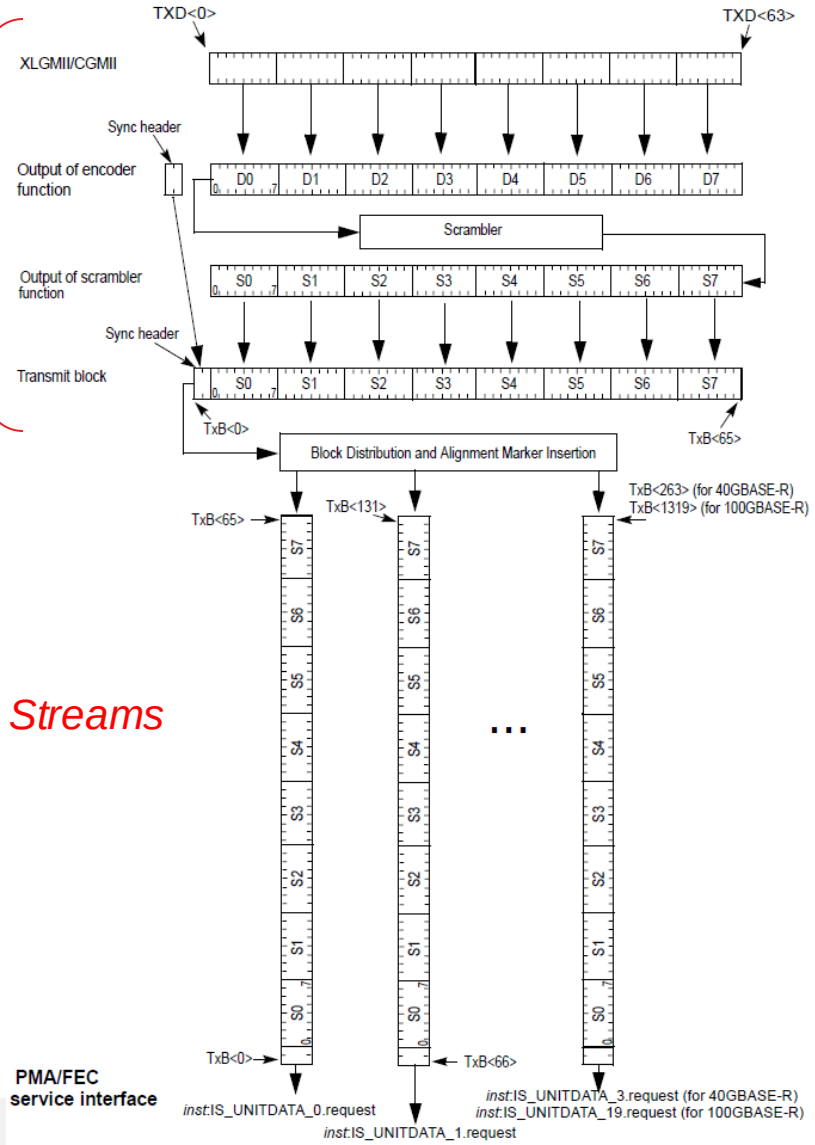


Figure 82–3—PCS Transmit bit ordering

66b Block Distribution

- Blocks are distributed in a “round robin” fashion

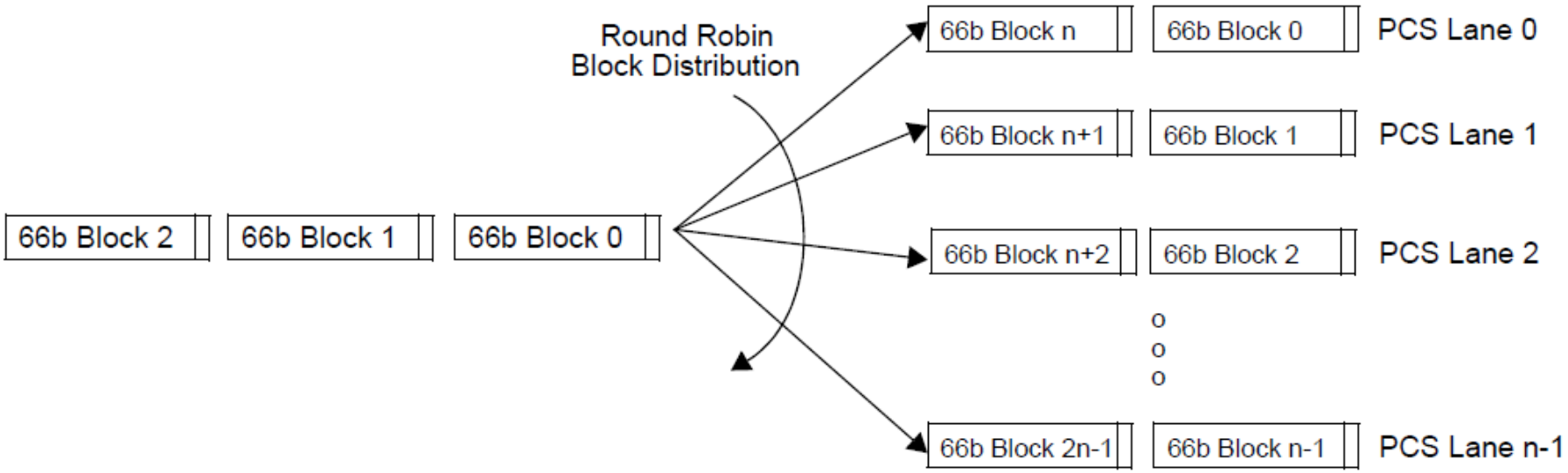


Figure 82–6—PCS Block distribution

Alignment Markers

- We need to deskew and reorder the individual PCS lanes at the receive PCS
- This is handled by inserting specially defined 66-bit blocks with a control block sync header

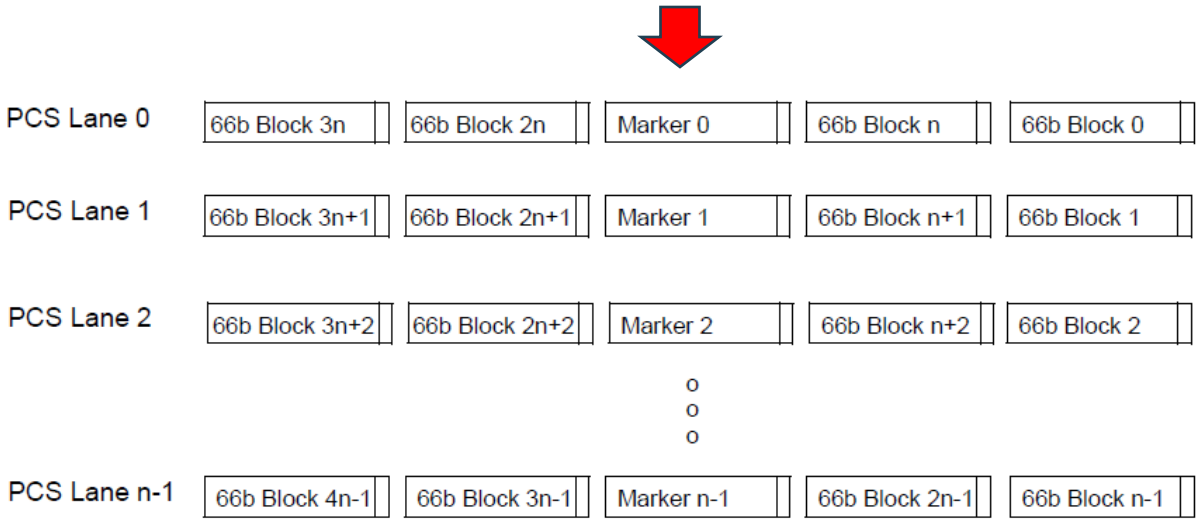


Figure 82–7—Alignment marker insertion

Alignment Marker Insertion Period (40GE, 100GE)

- Inserted after every 16383 66-bit Blocks on each PCS lane
  - Q. *Doesn't this increase the bandwidth ever so slightly?!?*
  - *A. No. Room for the AMs is created by periodically deleting IPG*
- not scrambled

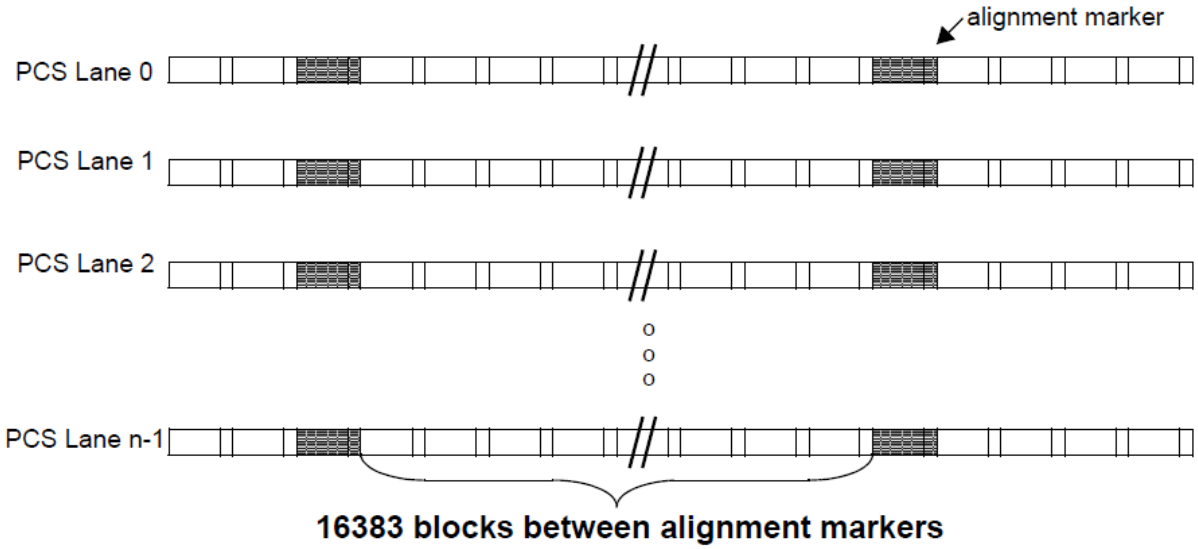


Figure 82–8—Alignment marker insertion period

Alignment Marker Format (40GE, 100GE)

- Sync header -> control type
- M0, M1, M2, M4, M5, M6 uniquely defined per PCS lane
- BIP3 and BIP7 parity check

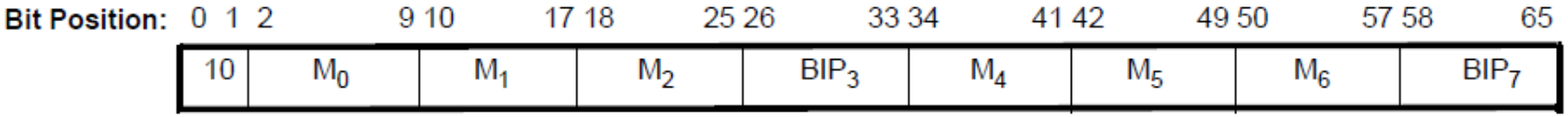


Figure 82–9—Alignment marker format

Alignment Marker Table 40GE

Table 82–3—40GBASE-R Alignment marker encodings

PCS lane number	Encoding[a] $\{M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7\}$
0	0x90, 0x76, 0x47, BIP_3, 0x6F, 0x89, 0xB8, BIP_7
1	0xF0, 0xC4, 0xE6, BIP_3, 0x0F, 0x3B, 0x19, BIP_7
2	0xC5, 0x65, 0x9B, BIP_3, 0x3A, 0x9A, 0x64, BIP_7
3	0xA2, 0x79, 0x3D, BIP_3, 0x5D, 0x86, 0xC2, BIP_7

Alignment Marker Table 100GE

Table 82–2—100GBASE-R Alignment marker encodings

PCS lane number	Encoding[a] $\{M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7\}$	PCS lane number	Encoding[a] $\{M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7\}$
0	0xC1, 0x68, 0x21, BIP_3, 0x3E, 0x97, 0xDE, BIP_7	10	0xFD, 0x6C, 0x99, BIP_3, 0x02, 0x93, 0x66, BIP_7
1	0x9D, 0x71, 0x8E, BIP_3, 0x62, 0x8E, 0x71, BIP_7	11	0xB9, 0x91, 0x55, BIP_3, 0x46, 0x6E, 0xAA, BIP_7
2	0x59, 0x4B, 0xE8, BIP_3, 0xA6, 0xB4, 0x17, BIP_7	12	0x5C, 0x B9, 0xB2, BIP_3, 0xA3, 0x46, 0x4D, BIP_7
3	0x4D, 0x95, 0x7B, BIP_3, 0xB2, 0x6A, 0x84, BIP_7	13	0x1A, 0xF8, 0xBD, BIP_3, 0xE5, 0x07, 0x42, BIP_7
4	0xF5, 0x07, 0x09, BIP_3, 0x0A, 0xF8, 0xF6, BIP_7	14	0x83, 0xC7, 0xCA, BIP_3, 0x7C, 0x38, 0x35, BIP_7
5	0xDD, 0x14, 0xC2, BIP_3, 0x22, 0xEB, 0x3D, BIP_7	15	0x35, 0x36, 0xCD, BIP_3, 0xCA, 0xC9, 0x32, BIP_7
6	0x9A, 0x4A, 0x26, BIP_3, 0x65, 0xB5, 0xD9, BIP_7	16	0xC4, 0x31, 0x4C, BIP_3, 0x3B, 0xCE, 0xB3, BIP_7
7	0x7B, 0x45, 0x66, BIP_3, 0x84, 0xBA, 0x99, BIP_7	17	0xAD, 0xD6, 0xB7, BIP_3, 0x52, 0x29, 0x48, BIP_7
8	0xA0, 0x24, 0x76, BIP_3, 0x5F, 0xDB, 0x89, BIP_7	18	0x5F, 0x66, 0x2A, BIP_3, 0xA0, 0x99, 0xD5, BIP_7
9	0x68, 0xC9, 0xFB, BIP_3, 0x97, 0x36, 0x04, BIP_7	19	0xC0, 0xF0, 0xE5, BIP_3, 0x3F, 0x0F, 0x1A, BIP_7

BIPx calculations for 40GE, 100GE

BIP3 = an even parity calculation over *all the previous bits in the PCS lane*, including the previous Alignment Marker.
BIP7 = bit-wise inversion of BIP3

The BIP fields also provide a method of estimating the BER. This is used for post-FEC BER estimation

50GE PCS

50GE PCS Protocol

50GE PCS Alignment Marker distribution is identical to 40GE PCS except:
- AMs are inserted every 20,479 Blocks instead of every 16,383

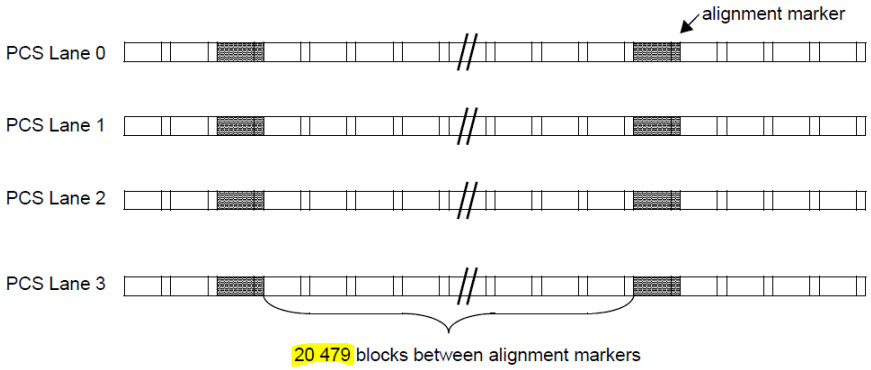


Figure 133–4—Alignment marker insertion period

Tighter Skew specs than 40/100GE:
MAX SKEW: 49 ns
MAX SKEW VARIATION: 0.4 ns

Q. What do you think happened before the 50GE protocol was ratified?
A. Manufacturers assumed the same AM block distribution spacing
=> **50GE Consortium mode**

PMA layer converts to 2 x 25G NRZ physical lanes

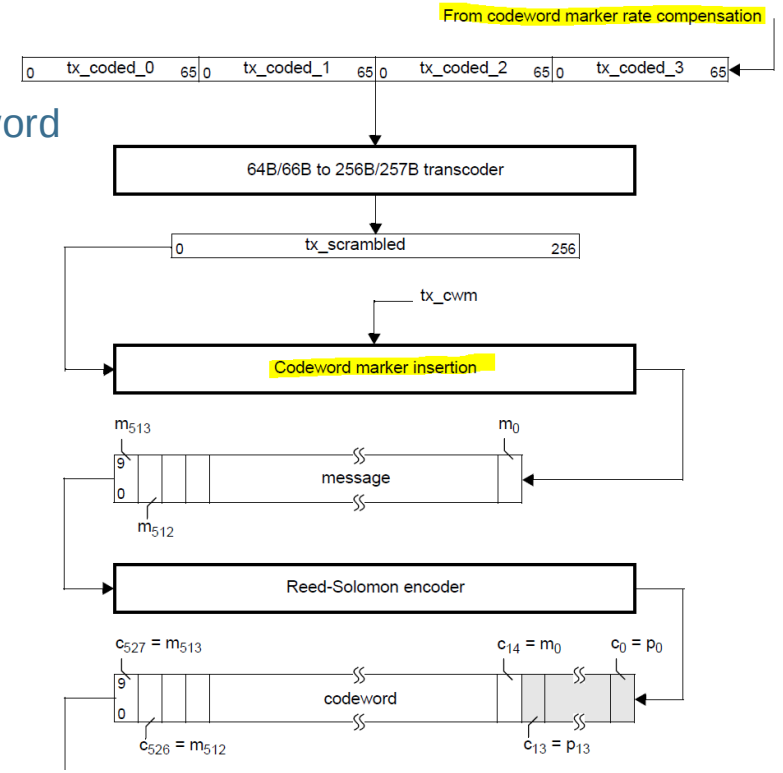
PCS Protocol Summary

Protocol	Num Lanes	Line Baud Rate Gbps	Alignment Markers
1GBASE-R	1	1.25	No
2.5BASE-R	1	3.125	No
10GBASE-			
10GBASE-KX4	4	3.125G	No
10GBASE-R	1	10.3125	No
25GBASE-R	1	25.78125	No
40GBASE-R4	4	10.3125	4
100GBASE-R			
100GBASE-R10	10	10.3125	20
100GBASE-R4	4	25.78125	20

RS(528, 514)

Codeword Markers for 25GBASE-R RS(528, 514)

- The receiving entity needs to be able to find the Codewords in the incoming data.
- To do this, the FEC encoder periodically inserts a **codeword marker**
- The codeword marker is a single 257-bit Block
- Inserted as the first 257 bits of every 1024th RS-FEC codeword
- distance between markers is every 20480 257-bit Block
- It is inserted BEFORE forming the codeword
- It does not get scrambled, so easy to find
- Room is created through rate compensation (IPG deletion)



Codeword Marker Formation for 25GBASE-R RS-528

- Reuses the encodings defined for 40GBASE-R Ethernet AM's
- These are 64b AM's, so take all 4 AM's to form one 257b Block
- No parity calculation for BIP3
  - BIP3 = 0x33, BIP7 = 0xCC
- Bit 257 set to 0

Table 82–3—40GBASE-R Alignment marker encodings

PCS lane number	Encoding[a] $\{M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7\}$
0	0x90, 0x76, 0x47, BIP_3, 0x6F, 0x89, 0xB8, BIP_7
1	0xF0, 0xC4, 0xE6, BIP_3, 0x0F, 0x3B, 0x19, BIP_7
2	0xC5, 0x65, 0x9B, BIP_3, 0x3A, 0x9A, 0x64, BIP_7
3	0xA2, 0x79, 0x3D, BIP_3, 0x5D, 0x86, 0xC2, BIP_7

[a]Each octet is transmitted LSB to MSB.

Full RS-528 25G Alignment Marker:

0 A2 79 3D 33 5D 86 C2 CC **C5 65 9B 33 3A 9A 64 CC** F0 C4 E6 33 0F 3B 19 CC **90 76 47 33 6F 89 B8 CC**

RS-528 for 100GBASE-KR4

- Only used for 4 x 25Gbps links (never 10 lane)
- Takes 100GBASE-R PCS and encodes with RS(528,514) FEC
- Translates the 20 PCS VLs to 4 FEC Lanes (or FECLs)
  - Each FECL AM is formed from 5 PCS AM's

PCS Lane, i
0, 4, 8, 12
1, 5, 9, 13, 17
2, 6, 10, 14, 18
3, 7, 11, 15, 19

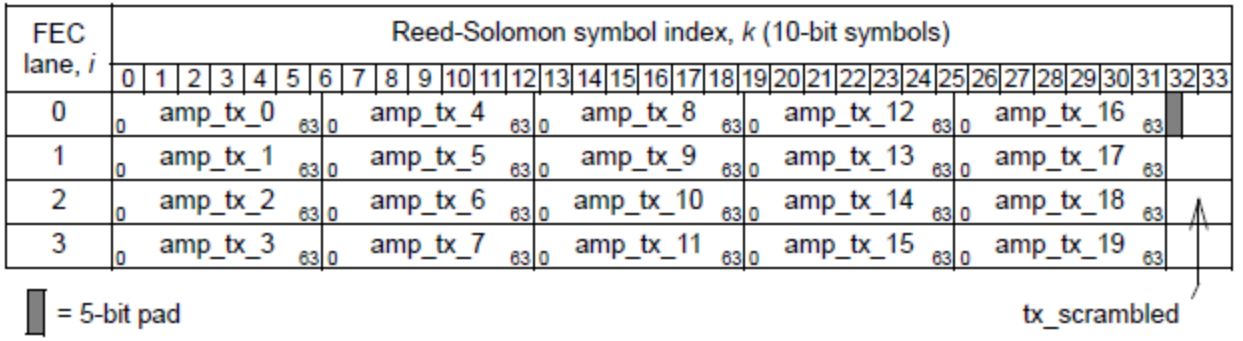
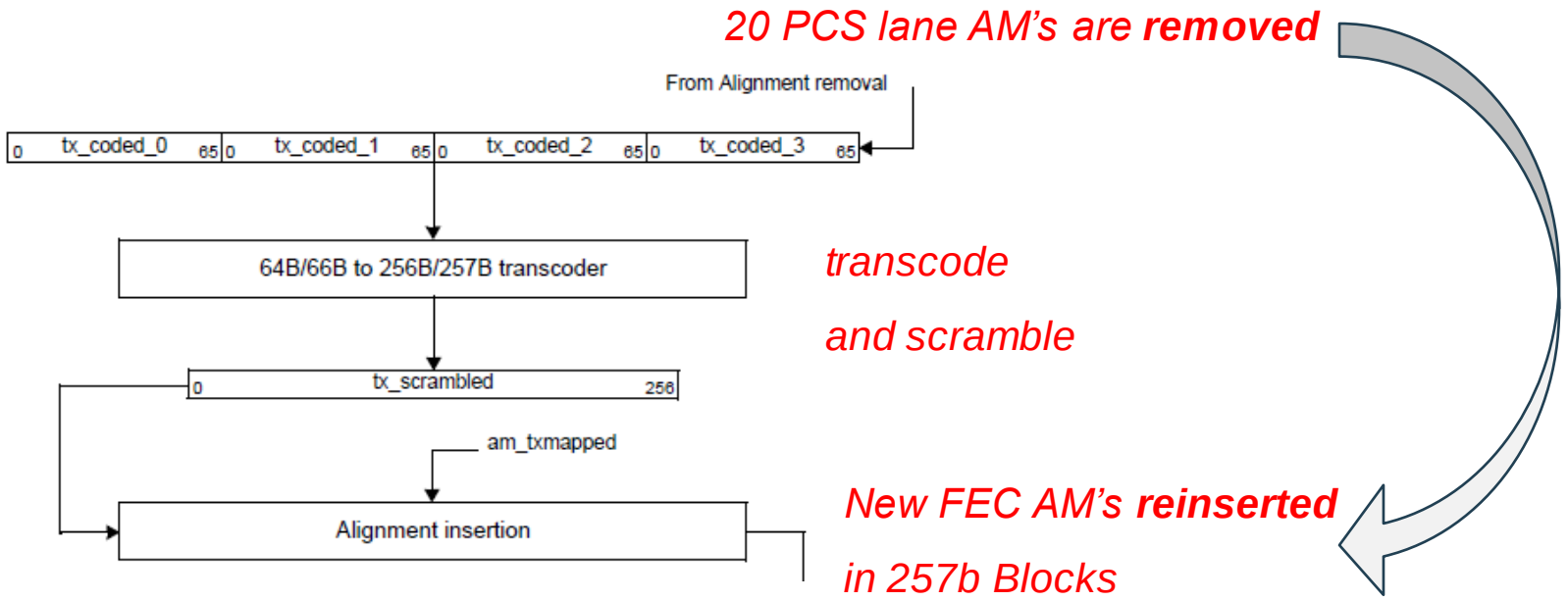


Figure 91–4—Alignment marker mapping to FEC lanes

RS-528 for 100GBASE-KR4



RS-528 for 100GBASE-KR4

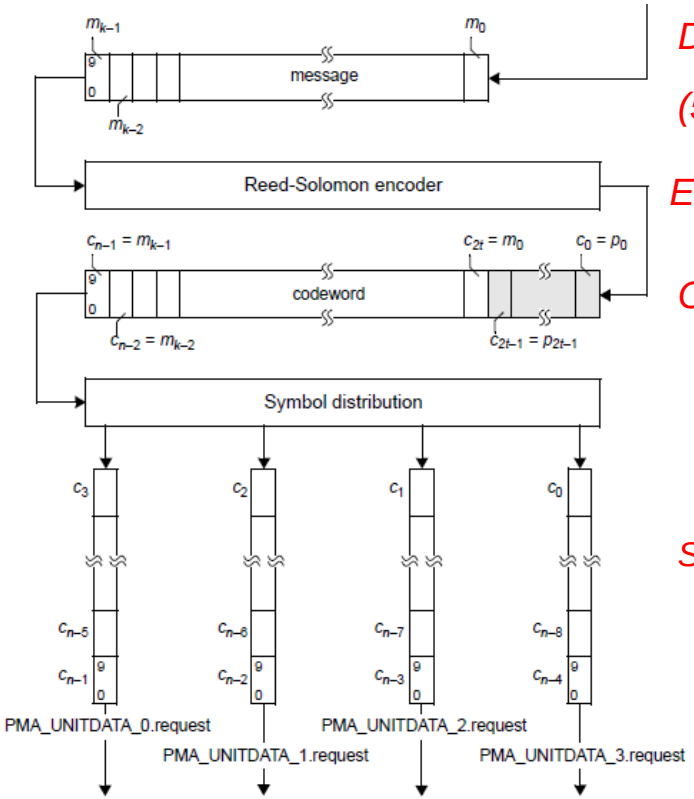


Figure 91–6—Transmit bit ordering

Data parsed into 5140 bit messages (514 symbols)

Encoder calculates 14-symbol parity

Codeword formed, 528 symbols

Symbols distributed cyclically (round robin)

4 FECLs

100GBASE-KP4 RS(544, 514)

- **Identical process to 100G RS-528**, except bigger parity block
- FECL bit rate = 26.5625Gbps

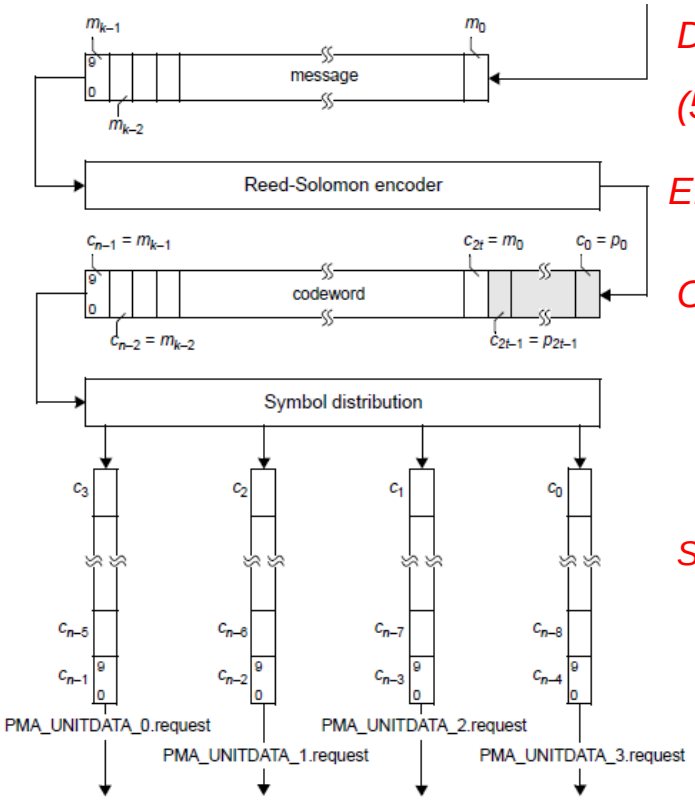


Figure 91–6—Transmit bit ordering

Data parsed into 5140 bit messages (514 symbols)

*Encoder calculates **30-symbol** parity*

*Codeword formed, **544** symbols*

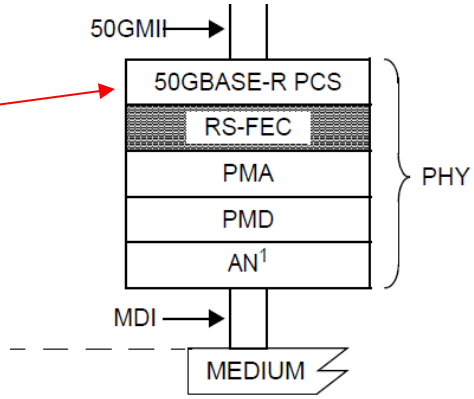
Symbols distributed cyclically (round robin)

4 FECLs

RS-544 FEC

RS-544 for 50GE

- Incoming 50GE is encoded as 4 x PCS virtual lanes with AM's
- PCS AM's are removed
- 66b Blocks transcoded to 257b Blocks (same as RS-528)
- The removed PCS AM's are reinserted after transcoding in a way such that they are mapped to **2 FEC virtual lanes**

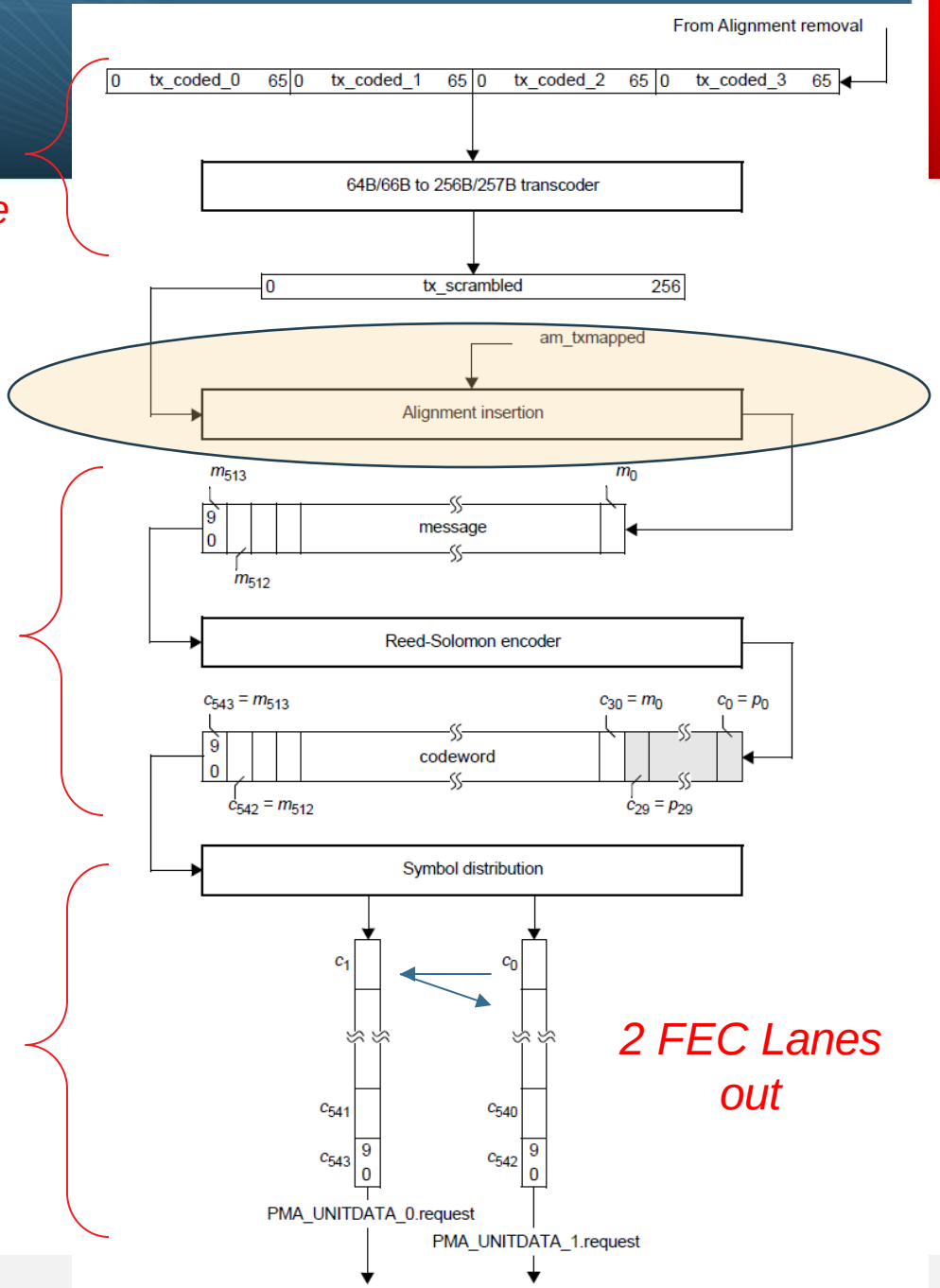


50G RS-544 Transmit

64b to 257b transcode

Codeword generation

- After FEC codeword formation, the data is passed as two FEC lanes (or FECLs) to the next layer
- Data is **symbol interleaved** to the two FECL's



2 FEC Lanes out

Figure 134–4—Transmit bit ordering

Codeword Marker Formation for 50GBASE-R RS-544

- 50G PCS uses 4 AM's (4 PCS virtual lanes), same AM's for 40G.
- 4 PCS AM's form one 257b Block
- Sequenced such that, *after* symbol distribution, you get:
  - Map PCS-AM0 + PCS-AM2 => FECL0
  - Map PCS-AM0 + PCS-AM3 => FECL1
  - Keep BIP3, BIP7 values from PCS

Table 82–3—40GBASE-R Alignment marker encodings

PCS lane number	Encoding[a] $\{M_0, M_1, M_2, BIP_3, M_4, M_5, M_6, BIP_7\}$
0	0x90, 0x76, 0x47, BIP_3, 0x6F, 0x89, 0xB8, BIP_7
1	0xF0, 0xC4, 0xE6, BIP_3, 0x0F, 0x3B, 0x19, BIP_7
2	0xC5, 0x65, 0x9B, BIP_3, 0x3A, 0x9A, 0x64, BIP_7
3	0xA2, 0x79, 0x3D, BIP_3, 0x5D, 0x86, 0xC2, BIP_7

[a]Each octet is transmitted LSB to MSB.

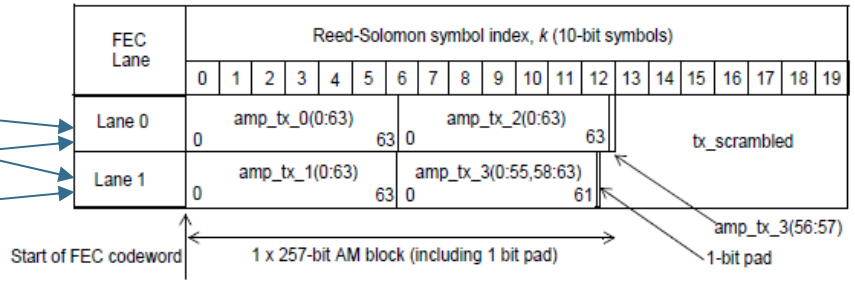
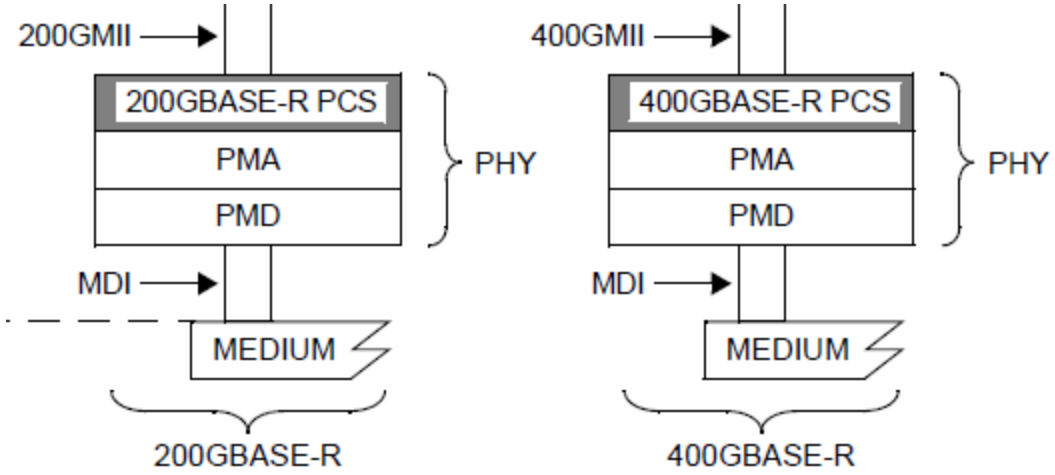


Figure 134–3—Alignment marker mapping to FEC lanes

- PCS AM0 is used for both FECLs.
  - Simplifies job of AM location in the receive decoder
  - After AM is located, use the second AM to determine which FECL it is

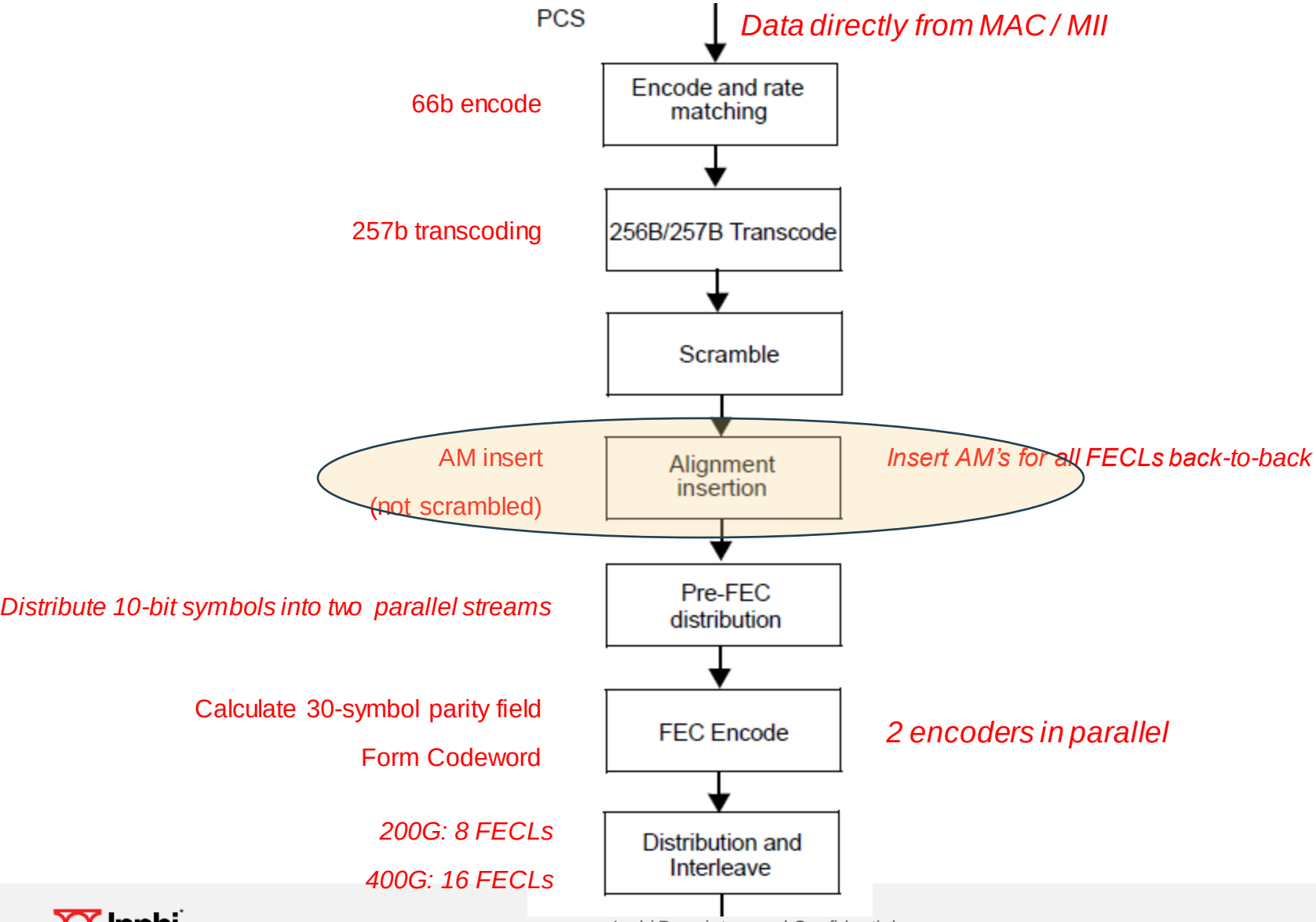
RS-544 for 200GBASE-R & 400GBASE-R

- **FEC is mandatory** at 200G, 400G
- Not a separate “layer”, built directly into the PCS



- **Two Reed-Solomon encoders** used, to handle the bandwidth
- Designed to accommodate 25G NRZ physical lanes (minimum)
  - For 200G, defines 8 FECLs
  - For 400G, defines 16 FECLs

RS-544 for 200GBASE-R & 400GBASE-R



200GBASE-R AM encodings

Table 119–1—200GBASE-R alignment marker encodings

PCS lane number	Encoding[a] $\{CM_0, CM_1, CM_2, UP_0, CM_3, CM_4, CM_5, UP_1, UM_0, UM_1, UM_2, UP_2, UM_3, UM_4, UM_5\}$
0	0x9A,0x4A,0x26,0x05,0x65,0xB5,0xD9,0xD6,0xB3,0xC0,0x8C,0x29,0x4C,0x3F,0x73
1	0x9A,0x4A,0x26,0x04,0x65,0xB5,0xD9,0x67,0x5A,0xDE,0x7E,0x98,0xA5,0x21,0x81
2	0x9A,0x4A,0x26,0x46,0x65,0xB5,0xD9,0xFE,0x3E,0xF3,0x56,0x01,0xC1,0x0C,0xA9
3	0x9A,0x4A,0x26,0x5A,0x65,0xB5,0xD9,0x84,0x86,0x80,0xD0,0x7B,0x79,0x7F,0x2F
4	0x9A,0x4A,0x26,0xE1,0x65,0xB5,0xD9,0x19,0x2A,0x51,0xF2,0xE6,0xD5,0xAE,0x0D
5	0x9A,0x4A,0x26,0xF2,0x65,0xB5,0xD9,0x4E,0x12,0x4F,0xD1,0xB1,0xED,0xB0,0x2E
6	0x9A,0x4A,0x26,0x3D,0x65,0xB5,0xD9,0xEE,0x42,0x9C,0xA1,0x11,0xBD,0x63,0x5E
7	0x9A,0x4A,0x26,0x22,0x65,0xB5,0xD9,0x32,0xD6,0x76,0x5B,0xCD,0x29,0x89,0xA4

First half is (mostly) the same, makes it easier to locate AM's in decoder.

Once AM location is found, use the rest of the AM to identify the lane number

400GBASE-R AM encodings

Table 119–2—400GBASE-R alignment marker encodings

PCS lane number	Encoding[a] $\{CM_0, CM_1, CM_2, UP_0, CM_3, CM_4, CM_5, UP_1, UM_0, UM_1, UM_2, UP_2, UM_3, UM_4, UM_5\}$
0	0x9A,0x4A,0x26,0xB6,0x65,0xB5,0xD9,0xD9,0x01,0x71,0xF3,0x26,0xFE,0x8E,0x0C
1	0x9A,0x4A,0x26,0x04,0x65,0xB5,0xD9,0x67,0x5A,0xDE,0x7E,0x98,0xA5,0x21,0x81
2	0x9A,0x4A,0x26,0x46,0x65,0xB5,0xD9,0xFE,0x3E,0xF3,0x56,0x01,0xC1,0x0C,0xA9
3	0x9A,0x4A,0x26,0x5A,0x65,0xB5,0xD9,0x84,0x86,0x80,0xD0,0x7B,0x79,0x7F,0x2F
4	0x9A,0x4A,0x26,0xE1,0x65,0xB5,0xD9,0x19,0x2A,0x51,0xF2,0xE6,0xD5,0xAE,0x0D
5	0x9A,0x4A,0x26,0xF2,0x65,0xB5,0xD9,0x4E,0x12,0x4F,0xD1,0xB1,0xED,0xB0,0x2E
6	0x9A,0x4A,0x26,0x3D,0x65,0xB5,0xD9,0xEE,0x42,0x9C,0xA1,0x11,0xBD,0x63,0x5E
7	0x9A,0x4A,0x26,0x22,0x65,0xB5,0xD9,0x32,0xD6,0x76,0x5B,0xCD,0x29,0x89,0xA4
8	0x9A,0x4A,0x26,0x60,0x65,0xB5,0xD9,0x9F,0xE1,0x73,0x75,0x60,0x1E,0x8C,0x8A
9	0x9A,0x4A,0x26,0x6B,0x65,0xB5,0xD9,0xA2,0x71,0xC4,0x3C,0x5D,0x8E,0x3B,0xC3
10	0x9A,0x4A,0x26,0xFA,0x65,0xB5,0xD9,0x04,0x95,0xEB,0xD8,0xFB,0x6A,0x14,0x27
11	0x9A,0x4A,0x26,0x6C,0x65,0xB5,0xD9,0x71,0x22,0x66,0x38,0x8E,0xDD,0x99,0xC7
12	0x9A,0x4A,0x26,0x18,0x65,0xB5,0xD9,0x5B,0xA2,0xF6,0x95,0xA4,0x5D,0x09,0x6A
13	0x9A,0x4A,0x26,0x14,0x65,0xB5,0xD9,0xCC,0x31,0x97,0xC3,0x33,0xCE,0x68,0x3C
14	0x9A,0x4A,0x26,0xD0,0x65,0xB5,0xD9,0xB1,0xCA,0xFB,0xA6,0x4E,0x35,0x04,0x59
15	0x9A,0x4A,0x26,0xB4,0x65,0xB5,0xD9,0x56,0xA6,0xBA,0x79,0xA9,0x59,0x45,0x86

First half is (mostly) the same AND IDENTICAL to 200G, makes it easier to locate AM's in decoder.

Once AM location is found, use the rest of the AM to identify the lane number

200GBASE-R Encoding

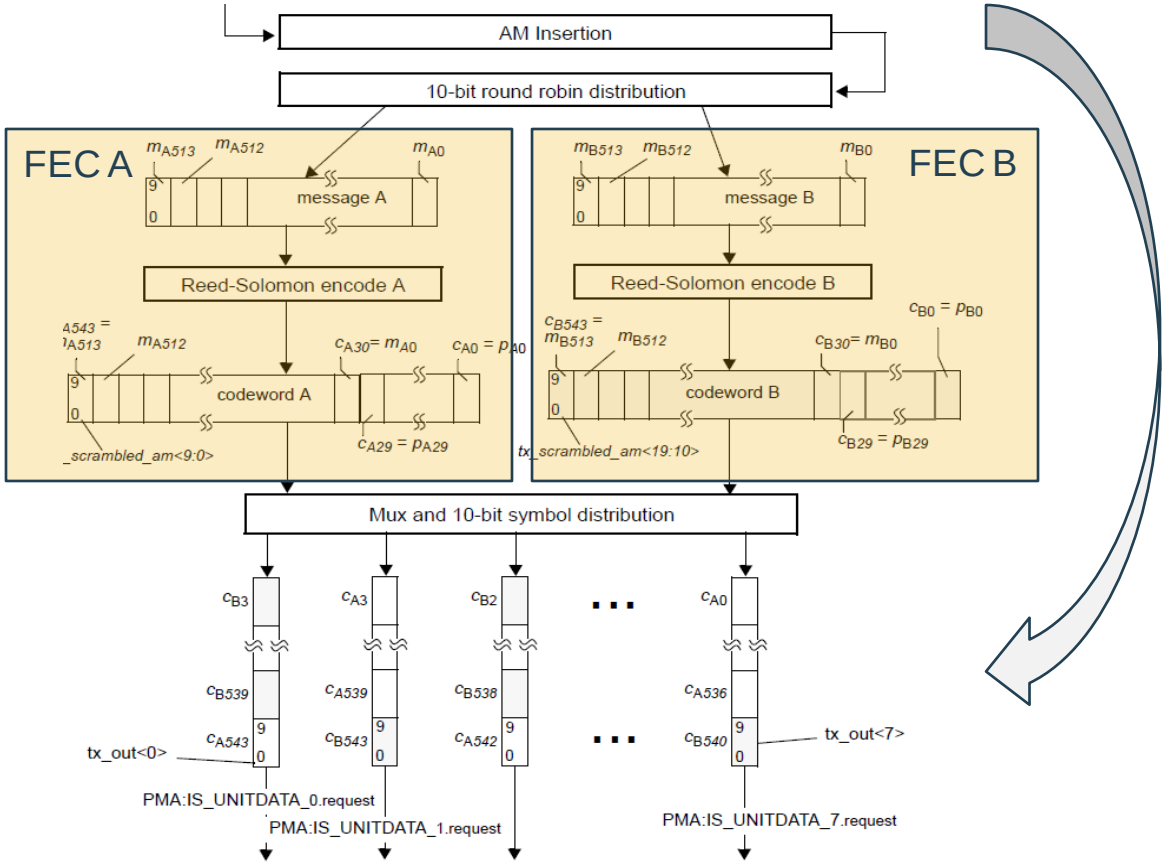


Figure 119–10—200GBASE-R Transmit bit ordering and distribution

Distribution done to ensure AM's wind up on correct FECL

400G is identical except 16 FECLs

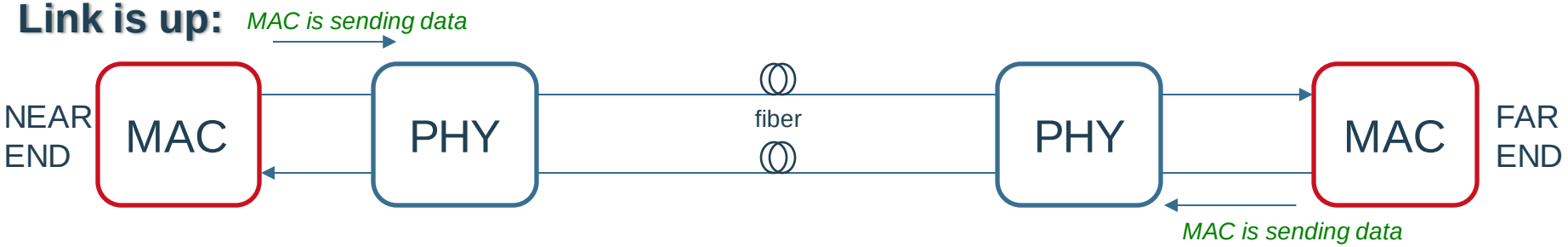
RS-FEC Protocol Summary

FEC	Rate	Num FECLs	Notes	IEEE Clause
RS-528	25GBASE-R	1	Uses Codeword Markers	
RS-528	100GBASE-KR4	4		802.3bj-2014 / 802.3-2015 Clause 91
RS-544	50GBASE-R	2		802.3cd-D3p5 Clause 134
RS-544	100GBASE-KP4	4		802.3bj-2014 / 802.3-2015 Clause 91
RS-544	200GBASE-R	8		802.3bs-2017 Clause 119
RS-544	400GBASE-R	16		802.3bs-2017 Clause 119

Link Fault Signalling

Link Fault Signalling

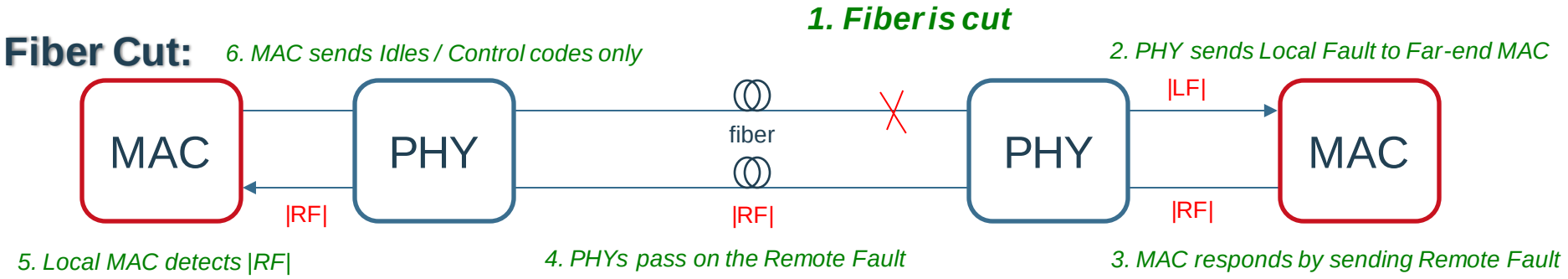
- Link Fault Signalling is a method of conveying “link down” information between MACs.
- In the Link Fault Signalling model, there are two basic elements: the PHY and the MAC



- When an Rx on a PHY fails, it can no longer transmit data to its link partner e.g. fiber cut. Instead, it transmits a Local Fault message |LF|.
- When Far-end MAC detects |LF|, it stops sending data and replaces with Remote Fault |RF|

Link Fault Signalling

- Near-end MAC detects |RF|, knows something is down. Stops sending data.
- When Source of problem clears (e.g. repair fiber cut), PHY stops sending |LF|. MAC then stops sending |RF| and both MACs will resume sending normal traffic
- => PHYs generate |LF|, MACs generate |RF|.
- => PHYs will pass both |LF| and |RF| transparently through them.



Link Fault Signalling - Encodings

Code	Ordered set	Number of code-groups	Encoding
||I||	**Idle**		**Substitute for XGMII Idle**
	Link Status		
||Q||	Sequence ordered set	4	/K28.4/Dx.y/Dx.y/Dx.y/[a]
||LF||	Local Fault signal	4	/K28.4/D0.0/D0.0/D1.0/
||RF||	Remote Fault signal	4	/K28.4/D0.0/D0.0/D2.0/

Code Group Name	Octet Value
D0.0	00
D1.0	01
D2.0	02
K28.3	7C
K28.4	9C

Octet Encodings

||I|| = 07 07 07 07

||LF|| = 9C 00 00 01

||RF|| = 9C 00 00 02