



FlexE 3.0 Implementation Agreement

IA # OIF-FLEXE-03.0a

February 2026

Implementation Agreement created and approved

OIF

www.oiforum.com



The OIF is an international non profit organization with over 160 member companies, including the world’s leading carriers and vendors. Being an industry group uniting representatives of the data and optical worlds, OIF’s purpose is to accelerate the deployment of interoperable, cost-effective and robust optical internetworks and their associated technologies. Optical internetworks are data networks composed of routers and data switches interconnected by optical networking elements.

With the goal of promoting worldwide compatibility of optical internetworking products, the OIF actively supports and extends the work of national and international standards bodies. Working relationships or formal liaisons have been established with EA, IEEE 802.3, INCITS T11, Infiniband, IPEC, ITU SG-15, PCI-SIG, SNIA SFF.

For additional information contact:

OIF

39221 Paseo Padre Pkwy, Suite J

Fremont, CA 94538 USA

+1.510.392.4903 □ info@oiforum.com

www.oiforum.com

Working Group: Physical and Link Layer

TITLE: FlexE 3.0 implementation agreement

SOURCE:

TECHNICAL EDITOR

Thomas J. Huber
Nokia
1305 Crestwood Court
Naperville, IL 60540 USA
Phone: +1 630 352 9005
Email: tom.huber@nokia.com

TECHNICAL EDITOR

Li Xu
Huawei Technologies Co., Ltd.
Bantian Street, Long Gang District
518129, ShenZhen, China
Phone: +86 15889304920
Email: innon.xu@huawei.com

WORKING GROUP CHAIR

David R. Stauffer, Ph.D.
Kandou Bus, S.A.
EPFL Innovation Park Bldg. I
1015 Lausanne Switzerland
Phone: +1 802 316 0808
Email: david@kandou.com

ABSTRACT: The Flex Ethernet (FlexE) Implementation Agreement provides a generic mechanism for supporting a variety of Ethernet MAC rates that may or may not correspond to any existing Ethernet PHY rate. This includes MAC rates that are both greater than (through bonding) and less than (through sub-rate and channelization) the Ethernet PHY rates used to carry FlexE. This can be viewed as a generalization of the Multi-Link Gearbox implementation agreements, removing the restrictions on the number of bonded PHYs (MLG2.0, for example, supports one or two 100GBASE-R PHYs) and the constraint that the FlexE Clients correspond to Ethernet rates (MLG2.0 supports only 10G and 40G clients).

FlexE 2.0 augments FlexE 1.0 by providing support for FlexE Groups composed of $n \times 200$ Gb/s Ethernet PHYs or of $n \times 400$ Gb/s Ethernet PHYs, and several other features. FlexE 2.1 augments FlexE 2.0 by providing support for FlexE Groups composed of $n \times 50$ Gb/s Ethernet PHYs. FlexE 2.2 is a maintenance release for FlexE 2.1.

FlexE 3.0 adds support for FlexE Groups composed of $n \times 800$ Gb/s Ethernet PHYs, payload type overhead, and 100G calendar slots..

Notice: This Technical Document has been created by the Optical Internetworking Forum (OIF). This document is offered to the OIF Membership solely as a basis for agreement and is not a binding proposal on the companies listed as resources above. The OIF reserves the



rights to at any time to add, amend, or withdraw statements contained herein. Nothing in this document is in any way binding on the OIF or any of its members.

The user's attention is called to the possibility that implementation of the OIF implementation agreement contained herein may require the use of inventions covered by the patent rights held by third parties. By publication of this OIF implementation agreement, the OIF makes no representation or warranty whatsoever, whether expressed or implied, that implementation of the specification will not infringe any third party rights, nor does the OIF make any representation or warranty whatsoever, whether expressed or implied, with respect to any claim that has been or may be asserted by any third party, the validity of any patent rights related to any such claim, or the extent to which a license to use any such rights may or may not be available or the terms hereof.

Copyright © 2024 Optical Internetworking Forum

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction other than the following, (1) the above copyright notice and this paragraph must be included on all such copies and derivative works, and (2) this document itself may not be modified in any way, such as by removing the copyright notice or references to the OIF, except as needed for the purpose of developing OIF Implementation Agreements.

By downloading, copying, or using this document in any manner, the user consents to the terms and conditions of this notice. Unless the terms and conditions of this notice are breached by the user, the limited permissions granted above are perpetual and will not be revoked by the OIF or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE OIF DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY, TITLE OR FITNESS FOR A PARTICULAR PURPOSE.

1 Table of Contents

1	TABLE OF CONTENTS	5
2	LIST OF FIGURES	7
3	LIST OF TABLES.....	9
4	DOCUMENT REVISION HISTORY	10
5	INTRODUCTION	11
5.1	Requirements	11
5.2	Relationship to IEEE 802.3 Stack.....	12
5.2.1	FlexE mux functions	12
5.2.2	FlexE Demux Functions	17
5.3	Sample Applications	22
6	GENERAL MECHANISM	25
6.1	FlexE Group.....	25
6.1.1	Groups composed of 50GBASE-R PHYs.....	26
6.1.2	Groups composed of 100GBASE-R PHYs.....	26
6.1.3	Groups composed of 200GBASE-R or 400GBASE-R PHYs	27
6.1.4	Groups composed of 800GBASE-R PHYs.....	27
6.2	FlexE Instances, padding and interleaving.....	28
6.3	Unequipped 100G FlexE Instances	30
6.4	FlexE Client rate adaptation	31
6.5	FlexE Calendar	31
6.6	FlexE Overhead and Alignment	33
7	DETAILED FUNCTIONS.....	38
7.1	FlexE Group Functions	38
7.2	FlexE Client Generation	38
7.2.1	FlexE Clients Generated internally within a system.....	38
7.2.2	FlexE Clients received from an Ethernet PHY.....	38
7.2.3	FlexE Clients from another FlexE Shim.....	40
7.2.4	Interconnect flexibility	40
7.3	FlexE Overhead Processing.....	40
7.3.1	FlexE Overhead Frame and Multiframe Lock.....	40
7.3.2	Calendar Configuration in Use	41

7.3.3	FlexE Map and FlexE Instance Number	41
7.3.4	Calendar Configuration	42
7.3.5	Management Channel(s) and Synchronization Messaging Channel.....	44
7.3.6	FlexE Group Number	46
7.3.7	Reserved Bits.....	46
7.3.8	Remote PHY Fault (RPF)	47
7.3.9	CRC-16.....	47
7.3.10	Payload Type	47
7.4	FlexE Mux Data Flow.....	48
7.5	FlexE Demux Data Flow	51
7.5.1	Skew Tolerance Requirements.....	53
7.5.2	FlexE Demux Fault Handling.....	53
7.6	FlexE Group Configuration.....	54
7.7	Energy Efficient Ethernet (EEE).....	54
7.8	FlexE test patterns	54
8	TRANSPORT NETWORK MAPPINGS FOR FLEX ETHERNET SIGNALS	55
8.1	FlexE Unaware Transport.....	55
8.2	FlexE termination in the Transport.....	55
8.3	FlexE Aware Transport.....	56
9	APPENDIX A: TEST VECTORS	58
10	APPENDIX B: (INFORMATIVE) ILLUSTRATION OF 25G CALENDAR SLOT DISTRIBUTION ACROSS 200G OR 400G PHYS.....	62
11	APPENDIX C: (INFORMATIVE) FLEXE CLIENT SYNCHRONIZATION.....	64
12	APPENDIX D: (INFORMATIVE) FLEXE AWARE CONFIGURATION EXAMPLE	65
13	REFERENCES	66
13.1	Normative references.....	66
14	APPENDIX E: LIST OF COMPANIES BELONGING TO OIF WHEN DOCUMENT IS APPROVED.....	66

2 List of Figures

Figure 1: General Structure of FlexE	11
Figure 2: 50GBASE-R FlexE mux functions	13
Figure 3: 100GBASE-R FlexE mux functions	13
Figure 4: 200GBASE-R FlexE mux functions	14
Figure 5: 400GBASE-R FlexE mux functions	14
Figure 6: 800GBASE-R FlexE mux functions	15
Figure 7: 50GBASE-R FlexE demux functions	18
Figure 8: 100GBASE-R FlexE demux functions	18
Figure 9: 200GBASE-R FlexE demux functions	19
Figure 10: 400GBASE-R FlexE demux functions	19
Figure 11: 800GBASE-R FlexE demux functions	20
Figure 12: Router to FlexE unaware Transport Network connection	23
Figure 13: FlexE terminating transport network equipment.....	24
Figure 14: Example of FlexE aware transport of Ethernet PHYs of a FlexE Group	25
Figure 15: Alignment of Overhead on Interleaved 100G FlexE Instances on 200GBASE-R, 400GBASE-R, and 800GBASE-R PHYs	29
Figure 16: Distribution of Pad blocks on 50GBASE-R, 200GBASE-R and 400GBASE-R PHYs.....	29
Figure 17: Distribution of Pad blocks on 800GBASE-R PHYs.....	30
Figure 18: Format of P1 Pad Block.....	30
Figure 19: Format of First Block of FlexE overhead frame for unequipped 100G FlexE Instances.....	30
Figure 20: Illustration of FlexE Calendar Distribution based on 5G granularity to length 10 sub-calendars (50G FlexE Instances).....	32
Figure 21: Illustration of FlexE Calendar Distribution based on 5G granularity (100G FlexE Instances)....	32
Figure 22: Illustration of FlexE calendar distribution based on 25G granularity	33
Figure 23: Illustration of FlexE calendar distribution based on 100G granularity	33
Figure 24 :Illustration of insertion of FlexE overhead on each 50G FlexE Instance of a FlexE Group (50GBASE-R PHYs only)	34
Figure 25:Illustration of insertion of FlexE overhead on each 100G FlexE Instance of a FlexE Group (100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R PHYs)	34
Figure 26: Illustration of Unavailable Calendar Slots on a 100G FlexE Instance to Facilitate Transport at lower rates	35
Figure 27: Encoding of Ordered Set block for FlexE overhead	35

Figure 28: FlexE Overhead Frame and Multiframe of each 50G FlexE Instance.....	37
Figure 29: FlexE Overhead Frame and Multiframe of each 100G FlexE Instance.....	37
Figure 30: Structure of FlexE Instance Number field.....	42
Figure 31: Ethernet Idle Control Block.....	45
Figure 32: Illustration of Data Flow for FlexE mux based on 50G FlexE Instances with length 10 sub-calendars.....	48
Figure 33: Illustration of Data Flow for FlexE Mux (5G calendar slots, 100G FlexE Instances)	49
Figure 34: Ethernet Error Control Block Format.....	49
Figure 35: Illustration of Data Flow for FlexE Mux (four 25G calendar slots per 100G FlexE Instance).....	50
Figure 36 : Illustration of Data Flow for FlexE Mux (single 100G calendar slot per 100G FlexE Instance) .	50
Figure 37: Illustration of FlexE Demux Data Flow (5G calendar slots from length 10 sub-calendars on 50G FlexE Instances).....	51
Figure 38: Illustration of FlexE Demux Data Flow (5G calendar slots, 100G FlexE Instances).....	51
Figure 39: Illustration of Data Flow for FlexE Demux (25G calendar slots from length 4 sub-calendars on 100G FlexE Instances)	52
Figure 40 : Illustration of Data Flow for FlexE Demux (100G calendar slots from length 1 sub-calendars on 100G FlexE Instances)	52
Figure 41: Ethernet Local Fault Ordered Set.....	54
Figure 42: Framed PRBS test pattern generator	55
Figure 43: Test Vector for first block of FlexE overhead frame	58
Figure 44: Test Vector for second block of FlexE overhead frame with PT codepoint 0x03	59
Figure 45: Test Vector for second block of FlexE overhead frame with PT codepoint 0x00	60
Figure 46: Test Vector for third block of FlexE overhead frame.....	61
Figure 47: Example of 25G Calendar Slots Multiplexed onto 200G FlexE PHYs	62
Figure 48: Example of 25G Calendar Slots Multiplexed onto a 400G FlexE PHY	62
Figure 49 : Example of 25G Calendar Slots Demultiplexed from 200G FlexE PHYs	63
Figure 50 : Example of 25G Calendar Slots Demultiplexed from a 400G FlexE PHY.....	63
Figure 51: Illustration of exchange of PTP event messages between MAC clients	64

3 List of Tables

Table 1: FlexE IA document revision history 10

Table 2: FlexE payload types 48

Table 3 – FlexE Aware Transport configuration example 65

4 Document Revision History

Table 1 provides the FlexE Implementation Agreement revision history.

Table 1: FlexE IA document revision history

Document	Date	Revisions/Comments
OIF-FLEXE-01.0	March 2016	Initial release
OIF-FLEXE-01.1	June 21, 2017	Maintenance Release: <ul style="list-style-type: none"> • Clarified bit transmission order • Added test vectors • Added example of using management channels to carry LLDP • Clarified 64b/66b encoding • Updated reference to G.709 • Clarified timer for calendar switch
OIF-FLEXE-02.0	June 22, 2018	Added support for groups composed of $n \times 200\text{G}$ or $n \times 400\text{G}$ PHYs, as well as other smaller features
OIF-FLEXE-02.1	July 2019	Added support for groups composed of $n \times 50\text{G}$ PHYs
OIF-FLEXE-02.2	October 2021	Maintenance release
OIF-FLEXE-03.0	May 2025	Added support for groups composed of $n \times 800\text{G}$ PHYs, new payload type overhead, and 100G calendar slots.
OIF-FLEXE-03.0a	February 2026	Corrected typographical errors in Figure numbers and cross-references in the published text of OIF-FLEXE-03.0

5 Introduction

The Flex Ethernet (FlexE) implementation agreement provides a generic mechanism for supporting a variety of Ethernet MAC rates that may or may not correspond to any existing Ethernet PHY rate. This includes MAC rates that are both greater than (through bonding) and less than (through sub-rate and channelization) the Ethernet PHY rates used to carry FlexE. This can be viewed as a generalization of the Multi-Link Gearbox implementation agreements, removing the restrictions on the number of bonded PHYs (MLG2.0, for example, supports one or two 100GBASE-R PHYs) and the constraint that the FlexE Clients correspond to Ethernet rates (MLG2.0 supports only 10G and 40G clients).

FlexE 2.1 augments FlexE 2.0 by providing support for FlexE Groups composed of 50 Gb/s Ethernet PHYs. FlexE 2.2 is a maintenance release for FlexE 2.1.

FlexE 3.0 adds support for FlexE Groups composed of 800 Gb/s Ethernet PHYs.

5.1 Requirements

The general capabilities supported by the FlexE implementation agreement are:

- Bonding of Ethernet PHYs, e.g., supporting a 200G MAC over two bonded 100GBASE-R PHYs.
- Sub-rates of Ethernet PHYs, e.g., supporting a 50G MAC over a 100GBASE-R PHY.
- Channelization within a PHY or a group of bonded PHYs, e.g, support a 150G and two 25G MACs over two bonded 100GBASE-R PHYs.

Note that hybrids are also possible, for example a sub-rate of a group of bonded PHYs, for example, a 250G MAC over three bonded 100GBASE-R PHYs.

The general approach is illustrated in Figure 1.

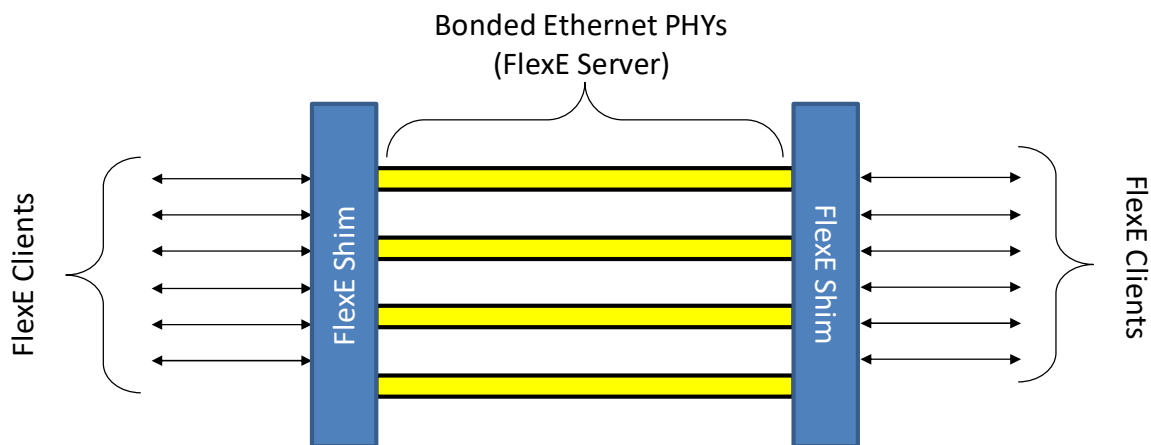


Figure 1: General Structure of FlexE

The *FlexE Group* consists of from 1 to m bonded Ethernet PHYs. This version of the Implementation Agreement supports FlexE Groups composed of 50GBASE-R PHYs, 100GBASE-R PHYs, 200GBASE-R PHYs, 400GBASE-R PHYs, or 800GBASE-R PHYs. All PHYs in the group must operate at the same rate.

The information carried across each PHY of the FlexE Group is described in terms of an information structure called a *FlexE Instance*. Two types of FlexE instances are defined: a 50G FlexE Instance and a 100G FlexE Instance. A group of 1 to m Ethernet PHYs carries from 1 to n FlexE Instances.

A 50G FlexE Instance is only carried over a 50GBASE-R PHY. A 100GBASE-R PHY carries a single 100G FlexE Instance. A 200GBASE-R PHY carries two 100G FlexE Instances. A 400GBASE-R PHY carries four 100G FlexE Instances. An 800GBASE-R PHY carries eight 100G FlexE Instances.

A *FlexE Instance* is a unit of information consisting of 50G or 100G of capacity able to carry FlexE Client data, together with its associated overhead. A FlexE Client is an Ethernet flow based on a MAC data rate that may or may not correspond to any Ethernet PHY rate. The FlexE Client MAC rates supported by FlexE Groups are 10, 40, and $m \times 25$ Gb/s. The FlexE Client MAC rates supported by FlexE Groups may support all, or only a subset of these FlexE Client rates, e.g., $m \times 25$ Gb/s or $p \times 100$ Gb/s. The logic pertaining to 50G FlexE Instances and 100G FlexE Instances is quite similar: whenever the term FlexE Instance is used in this implementation agreement without indication of a rate, the text applies to the processing of both 50G FlexE Instances and 100G FlexE Instances.

The *FlexE Shim* is the layer that maps or demaps the FlexE Clients carried over a FlexE Group. Similar to terminology of MLG, the *FlexE mux* refers to the transmit direction which maps the FlexE Clients over the FlexE Group. The *FlexE demux* refers to the receive direction which demaps the FlexE Clients from the FlexE Group.

5.2 Relationship to IEEE 802.3 Stack

The FlexE Shim can be envisioned as being in the middle of the PCS in the 50GBASE-R stack as illustrated in [802.3] Figure 131-1, the 100GBASE-R stack as illustrated in [802.3] Figure 80-1, the 200GBASE-R or 400GBASE-R stack as illustrated in [802.3] Figure 116-1, or the 800GBASE-R stack as illustrated in [802.3df] Figure 169-1.

Each FlexE Client has its own separate MAC, Reconciliation Sublayer, and xMII above the FlexE Shim which operate at the FlexE Client rate. The layers below the PCS (e.g., 100GBASE-R PMA, optional FEC, PMD) are used intact as specified for Ethernet.

5.2.1 FlexE mux functions

The functions of the FlexE mux (the FlexE Shim functions in the transmit direction) are illustrated in Figure 2 for FlexE groups composed of 50GBASE-R PHYs, in Figure 3 for FlexE Groups composed of 100GBASE-R PHYs, in Figure 4 for FlexE groups composed of 200GBASE-R PHYs, in Figure 5 for FlexE Groups composed of 400GBASE-R PHYs, and in Figure 6 for FlexE Groups composed of 800GBASE-R PHYs.

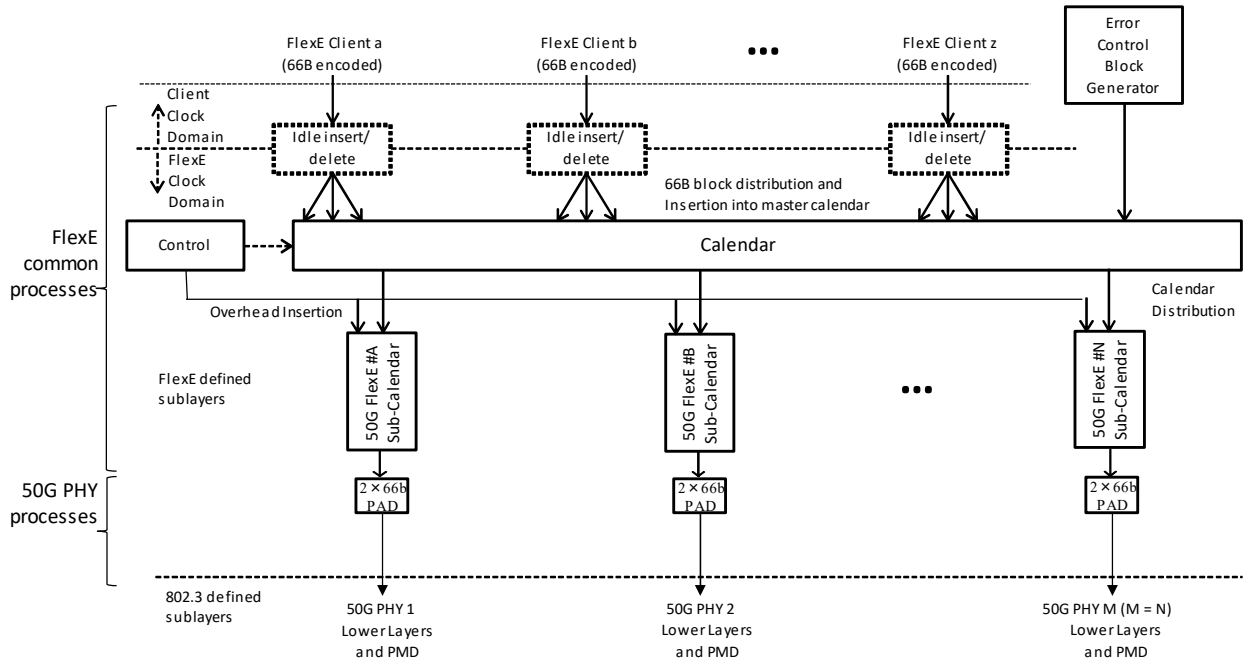


Figure 2: 50GBASE-R FlexE mux functions

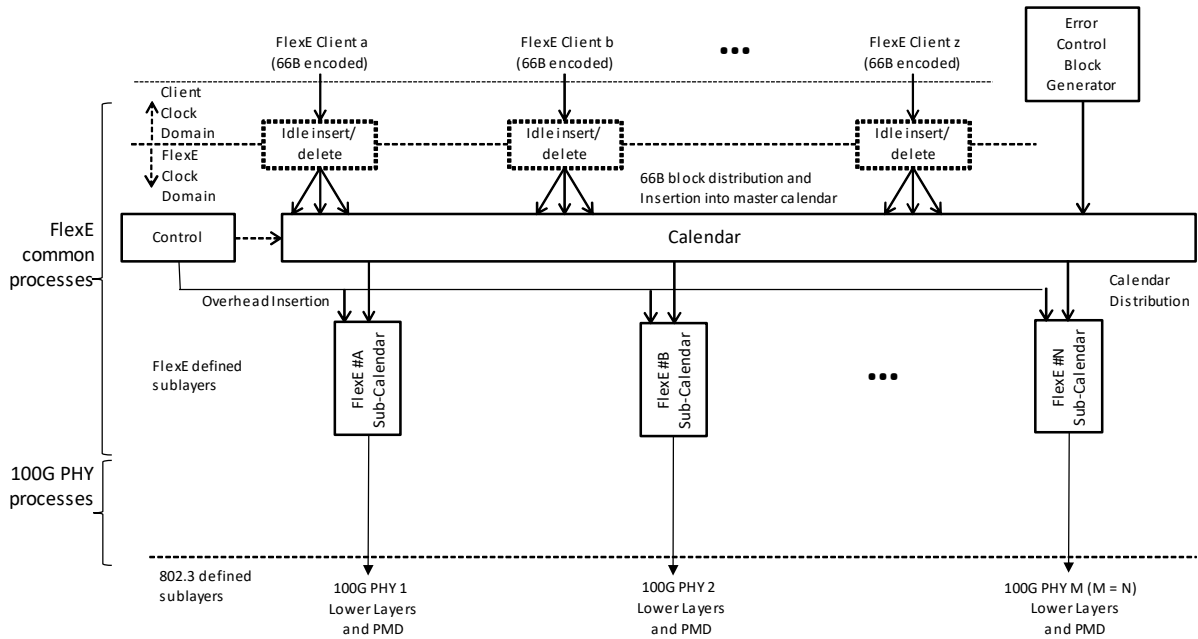


Figure 3: 100GBASE-R FlexE mux functions

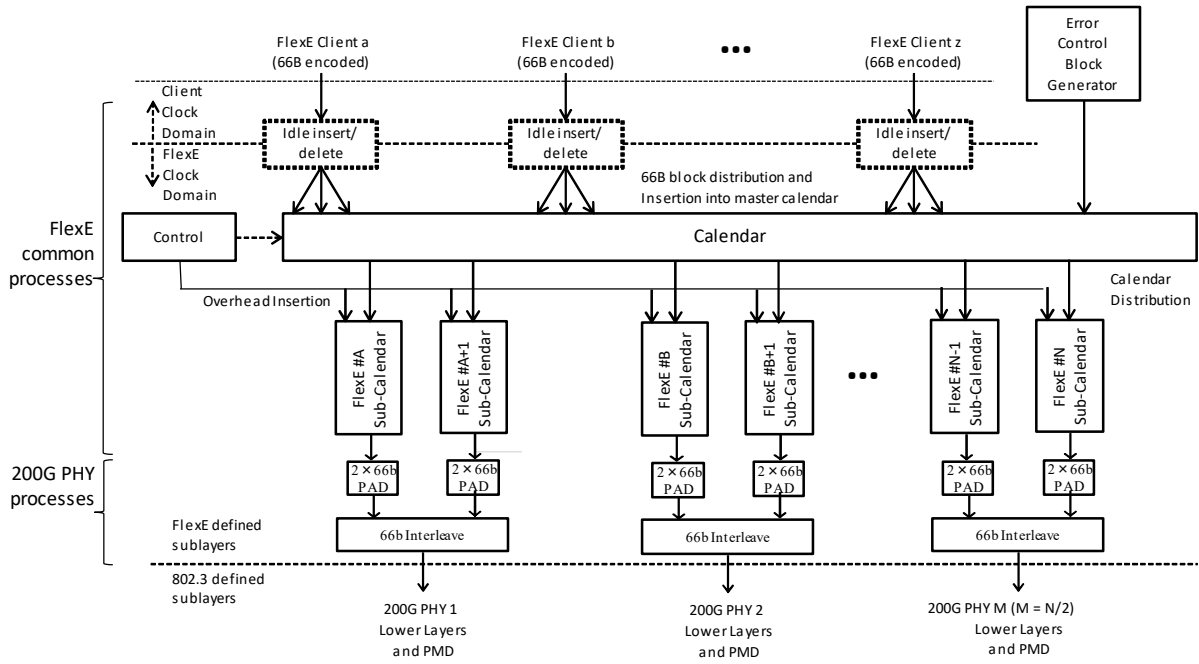


Figure 4: 200GBASE-R FlexE mux functions

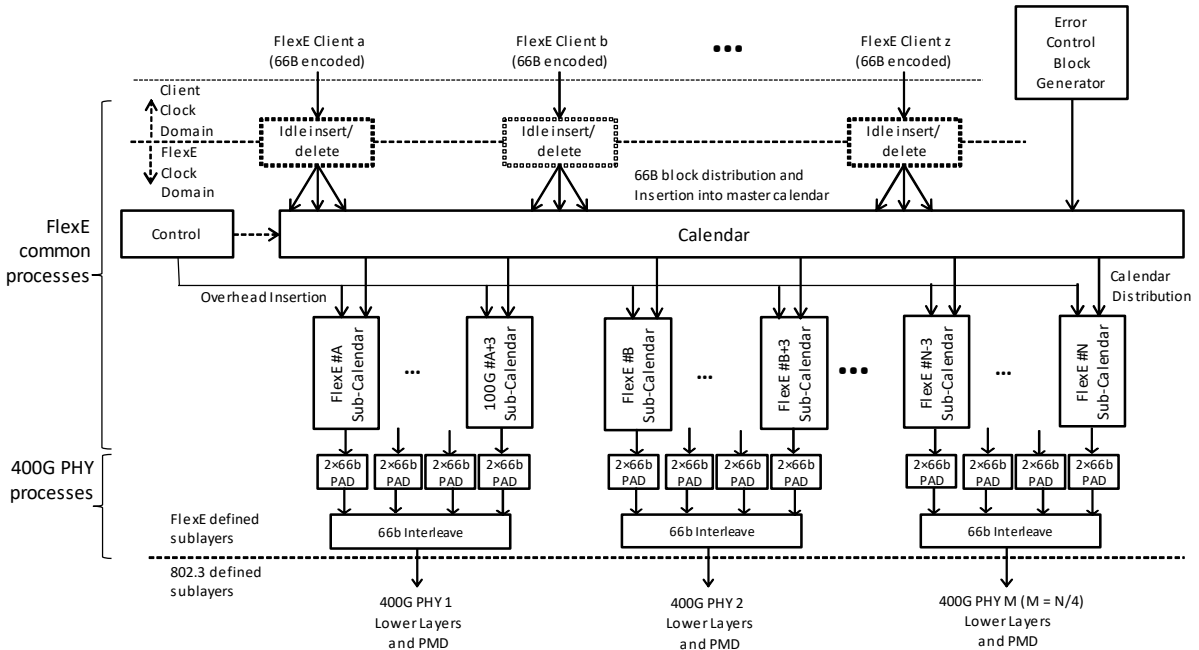


Figure 5: 400GBASE-R FlexE mux functions

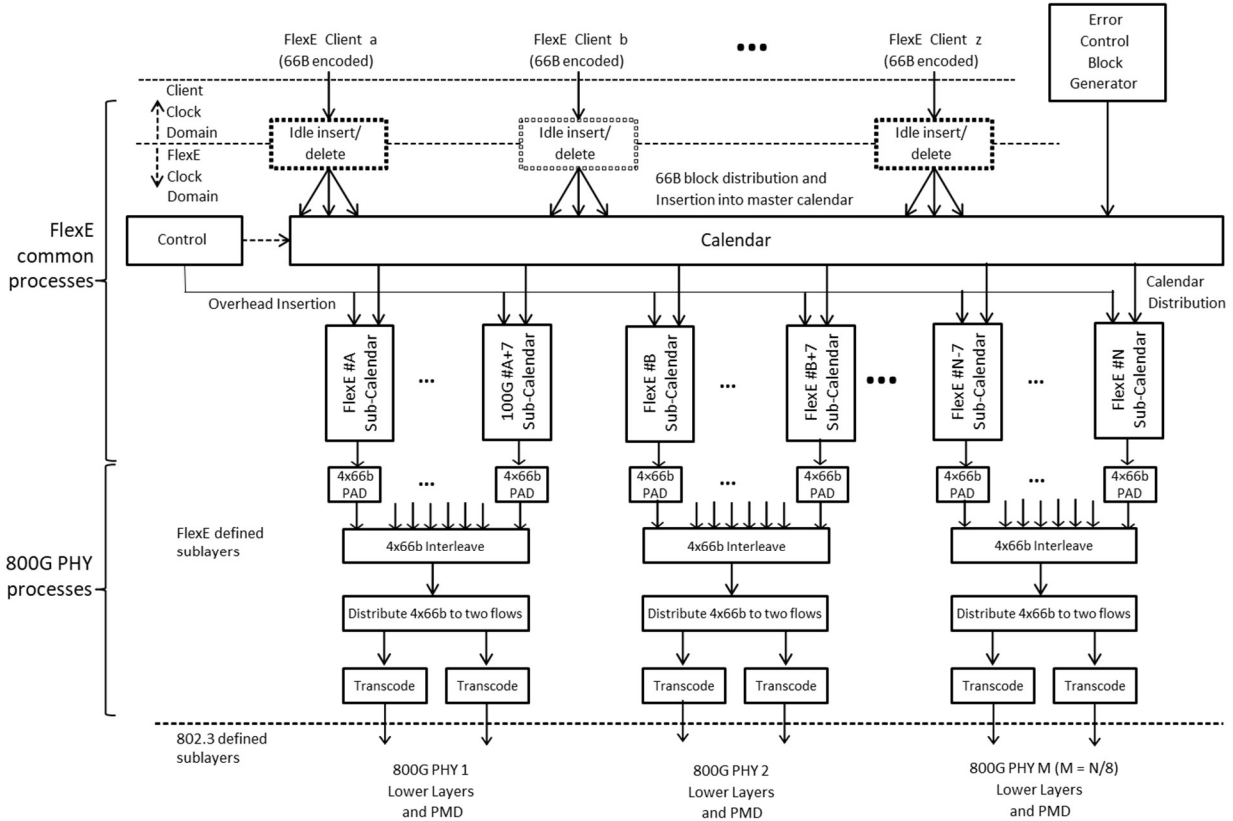


Figure 6: 800GBASE-R FlexE mux functions

5.2.1.1 FlexE Client

Each FlexE Client is presented to the FlexE Shim as a 64B/66B encoded bit-stream according to [802.3] Figure 82-5 or [802.3df] Table 172-1 stateless encoder rules. How this bit-stream is created is application specific (see clause 7.2 for details), but it should appear to the FlexE Shim as having been created from an Ethernet MAC operating at a rate of 10, 40, or $m \times 25$ Gb/s (or a subset of these rates, e.g., $m \times 25$ Gb/s or $p \times 100$ Gb/s), through a logical RS layer which performs the link fault signaling functions described in [802.3] clause 81.3.4. The content of FlexE Client stream may be LF in the case of an upstream failure of the FlexE Client. The start control character is aligned to an 8-byte boundary, feeding through a logical xMII at the corresponding rate, and a 64B/66B encoder resulting in a FlexE Client rate of:

$$\frac{66}{64} \times \text{FlexE Client MAC rate} \pm \text{bit rate tolerance}$$

The bit rate tolerance for FlexE Clients is ± 100 ppm. In the case of an 800G FlexE Client being carried over an 800GBASE-R PHY, the tolerance could be reduced to ± 50 ppm.

Note – In cases of connecting a FlexE client from one group to another without recoding or interworking from an Ethernet PHY to a FlexE client without recoding, it is advisable to replace any block with a corrupted sync header (which may have resulted from error marking by an upstream FEC decoder) with an error control block (see Figure 15) prior to insertion in the FlexE calendar.

5.2.1.2 Idle insert/delete

All FlexE Clients must be rate-adapted to match the clock of the FlexE Group. This is accomplished by idle insertion/deletion according to [802.3] clause 82.2.3.6 and/or ordered set deletion according to [802.3] clause 82.2.3.9. The nominal rate of the adapted signal is slightly less than the nominal rate of the FlexE Client to allow room for the alignment markers on the PHYs of the FlexE Group and insertion of the FlexE overhead.

Note – Idle insertion/deletion is appropriate when the nominal rate of the FlexE Client is the same as the nominal rate of the calendar slots into which it is mapped, differing by no more than a few hundred ppm based on the presence of alignment markers, FlexE overhead, and pad blocks in addition to the ± 100 ppm possible variation between the clocks of different Ethernet physical interfaces. Idle insertion/deletion cannot be used to bridge large differences between the rate of a FlexE Client and the calendar slot capacity, e.g., to map a 10G FlexE Client into a 25G calendar slot. Such implementations would require a MAC frame buffer.

5.2.1.3 66B Block Distribution and Insertion into Calendar

The 66B blocks from each FlexE Client are distributed sequentially into the calendar in the order described in clause 6.5.

5.2.1.4 Calendar Distribution

The 66B blocks from calendar are distributed to each PHY of the FlexE Group according to the ordering described in clause 6.5. The FlexE overhead is inserted into the sub-calendar of each PHY as described in clause 7.4.

5.2.1.5 Padding, Interleaving and distribution to the PHY functions

For the case of 200G, 400G and 800G FlexE PHYs, the p 100G FlexE Instances are padded and interleaved as described in clause 6.2.

Per [802.3df], the 800GBASE-R PCS is composed of two 400G flows identified as flow 0 and flow 1. For the case of 800G FlexE PHYs, the interleaving of the eight 100G FlexE Instances is done in groups of four 66b blocks, and the FlexE mux includes the processes of distributing groups of four 66b blocks to two 400G flows and 256B/257B transcoding that transcodes each set of four 66b blocks as specified in clause 172.2.4.4 of [802.3df]. The distribution process is aligned to the FlexE frame such that each 257b block produced by the transcoder contains four 66b blocks from the same FlexE Instance. Further, it is aligned such that FlexE instances 0, 2, 4, and 6 are distributed to flow 0 and FlexE instances 1, 3, 5, and 7 are distributed to flow 1.

5.2.1.6 PHY functions (Scramble, lane distribution, AM insertion, PMA, FEC, PMD)

5.2.1.6.1 50GBASE-R PHY functions (Scramble, lane distribution, AM insertion, PMA, FEC, PMD)

The stream of 66B blocks of each PHY is distributed to the PCS lanes of that PHY with insertion of alignment markers, and this is presented at the PMA service interface in the 50GBASE-R stack. Lower layers and interfaces of the 50GBASE-R Ethernet PHY (e.g., LAUI/50GAUI, FEC, PMA, PMD) are used as specified in [802.3].

5.2.1.6.2 100GBASE-R PHY functions (Scramble, lane distribution, AM insertion, PMA, FEC, PMD)

The stream of 66B blocks of each PHY is distributed to the PCS lanes of that PHY with insertion of alignment markers, and this is presented at the PMA service interface in the 100GBASE-R stack. Lower layers and interfaces of the 100GBASE-R Ethernet PHY (e.g., CAUI, FEC, PMA, PMD) are used as specified in [802.3].

5.2.1.6.3 200GBASE-R and 400GBASE-R PHY functions (Scramble, lane distribution, AM insertion, FEC, PMA, PMD)

The stream of 66B blocks of each PHY is presented at the input of the 257B transcoder as shown in [802.3] Figure 119-2, so it will be 257B transcoded, scrambled, FEC encoded, distributed to PCS lanes, and alignment markers inserted for presentation at the PMA service interface in the 200GBASE-R or 400GBASE-R stack. Lower layers and interfaces of 200GBASE-R and 400GBASE-R PHYs (e.g., 200G/400GAUI, PMA, PMD) are used as specified in [802.3].

5.2.1.6.4 800GBASE-R PHY functions (Distribute to flows, transcode, scramble, AM insertion, lane distribution, FEC, PMA, PMD)

The two 400G flows of 257B blocks of each PHY are presented at the input of the scrambler as shown in [802.3df] Figure 172-2. Each flow will be scrambled, FEC encoded, distributed to PCS lanes, and alignment markers inserted for presentation at the PMA service interface in the 800GBASE-R stack. Lower layers and interfaces of 800GBASE-R PHYs (e.g., 800GAUI, PMA, PMD) are used as specified in [802.3df].

5.2.1.7 Error Control Block Generator

Error Control blocks are generated for insertion into calendar slots that are unused or unavailable. See Figure 35.

5.2.1.8 Control

The control function manages the calendar slots into which each FlexE Client is inserted and inserts the FlexE overhead on each FlexE PHY in the transmit direction.

5.2.2 FlexE Demux Functions

The functions of the FlexE demux (the FlexE Shim in the receive direction) are illustrated in Figure 7 for FlexE Groups composed of 50GBASE-R PHYs, in Figure 8 for FlexE Groups composed of 100GBASE-R PHYs, in Figure 9 for FlexE Groups composed of 200GBASE-R PHYs, in Figure 10 for FlexE Groups composed of 400GBASE-R PHYs, and in Figure 11 for FlexE Groups composed of 800GBASE-R PHYs.

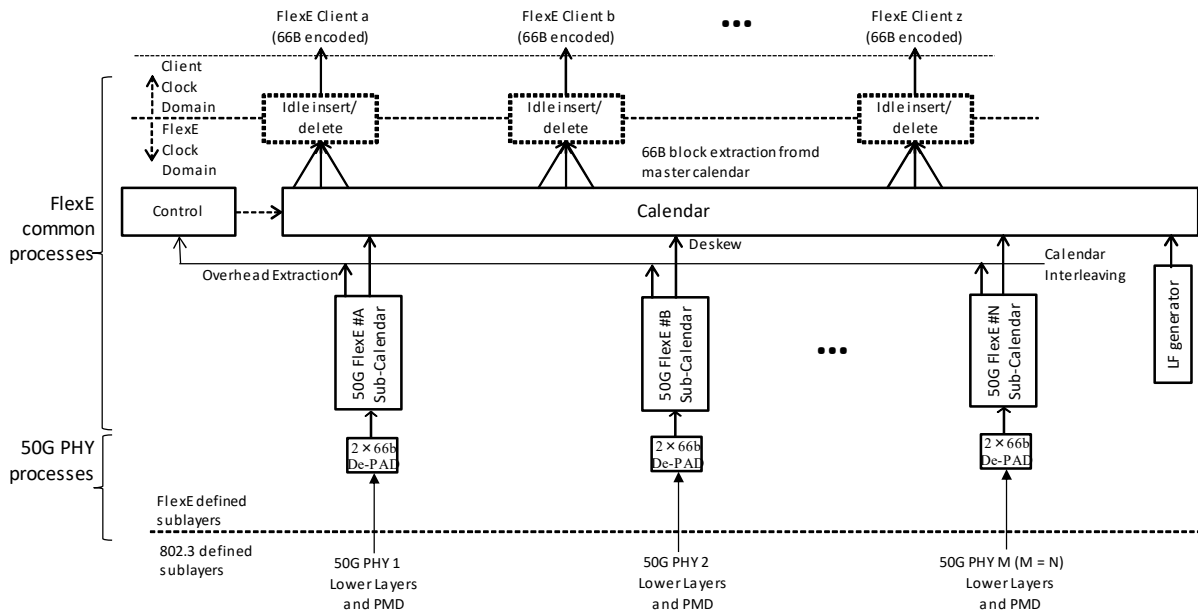


Figure 7: 50GBASE-R FlexE demux functions

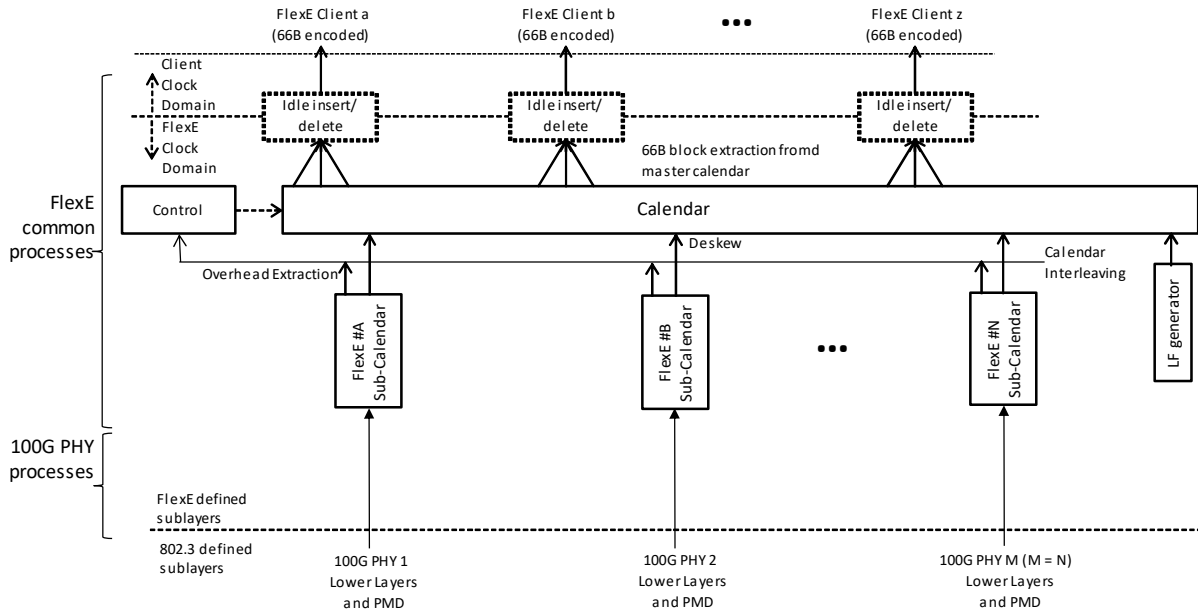


Figure 8: 100GBASE-R FlexE demux functions

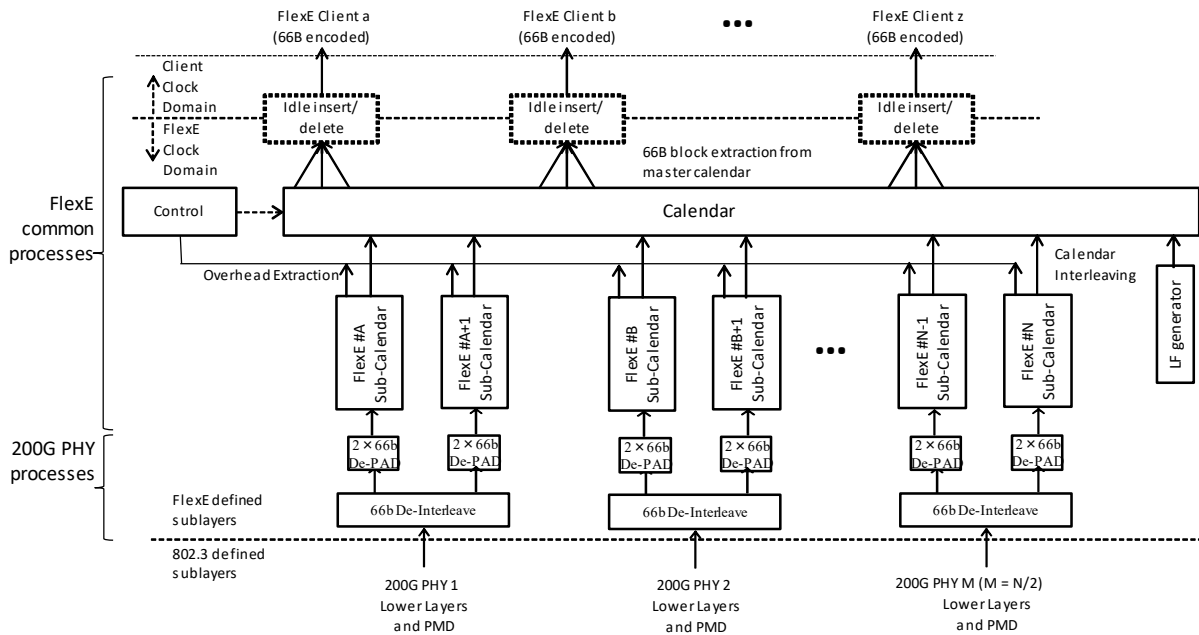


Figure 9: 200GBASE-R FlexE demux functions

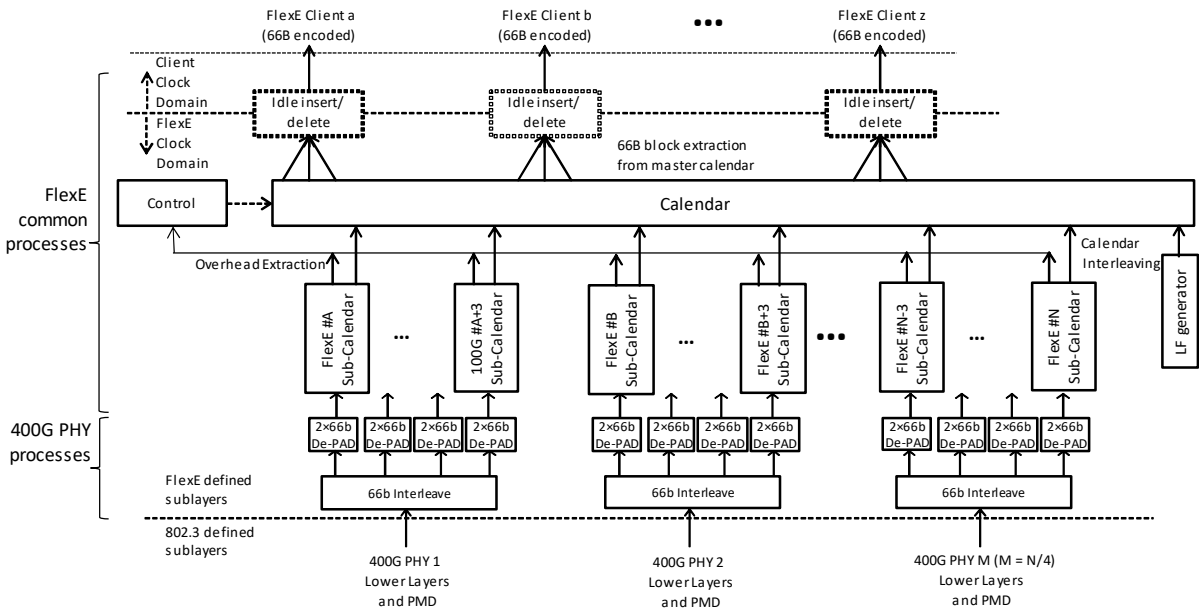


Figure 10: 400GBASE-R FlexE demux functions

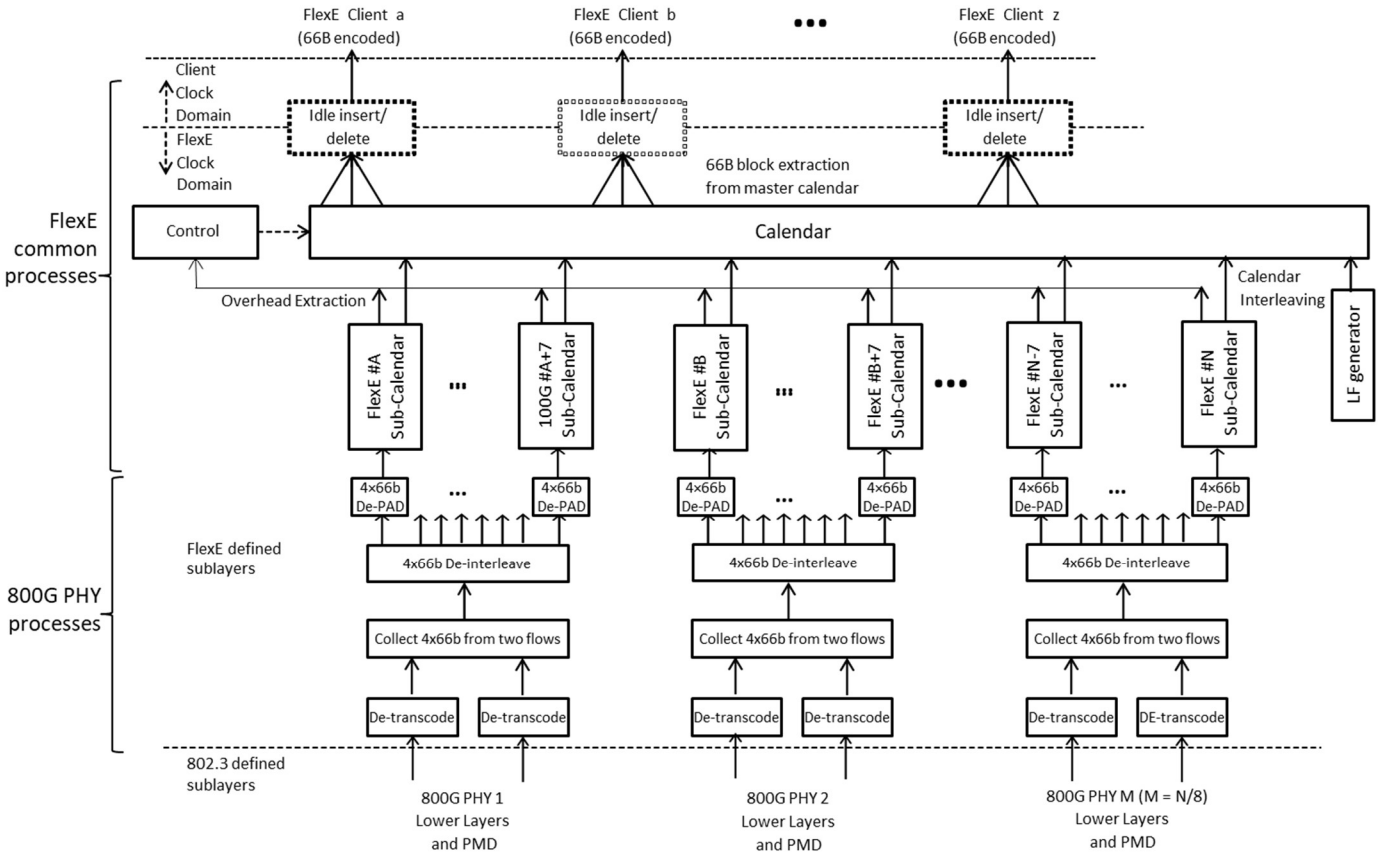


Figure 11: 800GBASE-R FlexE demux functions

5.2.2.1 PHY functions (PMD, PMA, FEC, lane deskew, interleave, AM removal, descramble)

5.2.2.1.1 50GBASE-R PHY functions (PMD, PMA, FEC, lane deskew, interleave, AM removal, descramble)

The layers of each 50GBASE-R PHY below the PCS are used exactly as specified in [802.3]. The PCS lanes are recovered, deskewed, reinterleaved, and the alignment markers are removed. The aggregate stream is descrambled. A 50GBASE-R PHY carries one 50G FlexE Instance.

Note – When FlexE groups composed of 50GBASE-R PHYs are used for channelized or subrated interfaces, it may be advisable to modify the error marking done in the FEC decoder to be more appropriate to a calendar slot structure to maintain satisfactory mean-time-to-false-packet acceptance (MTTFPA) on all FlexE clients.

5.2.2.1.2 100GBASE-R PHY functions (PMD, PMA, FEC, lane deskew, interleave, AM removal, descramble)

The layers of each 100GBASE-R PHY below the PCS are used exactly as specified in [802.3]. The PCS lanes are recovered, deskewed, reinterleaved, and the alignment markers are removed. The aggregate stream is descrambled. A 100GBASE-R PHY carries one 100G FlexE Instance.

Note – When FlexE groups composed of 100GBASE-R PHYs using RS-FEC are used for channelized or subrated interfaces, it may be advisable to modify the error marking done in the FEC decoder to be more appropriate to a calendar slot structure to maintain satisfactory mean-time-to-false-packet acceptance (MTTFPA) on all FlexE clients.

5.2.2.1.3 200GBASE-R and 400G BASE-R PHY functions (PMD, PMA, lane deskew, interleave, FEC decode, AM removal, descramble, reverse transcode)

The layers of each 200GBASE-R or 400GBASE-R PHY below the PCS are exactly as specified in [802.3]. The PCS lanes are recovered, deskewed, reordered, de-interleaved, FEC decoded, and post-FEC interleaved. The alignment markers are removed, the stream is descrambled, and reverse-transcoded to 66B format. A 200GBASE-R PHY can carry two 100G FlexE Instances, and a 400GBASE-R can carry four 100G FlexE Instances.

5.2.2.1.4 800GBASE-R PHY functions (PMD, PMA, lane deskew, interleave, FEC decode, AM removal, descramble, reverse transcode, collect)

The layers of each 800GBASE-R PHY below the PCS are exactly as specified in [802.3df]. The PCS lanes are recovered, deskewed, reordered, de-interleaved, FEC decoded, and post-FEC interleaved. The alignment markers are removed, the stream is descrambled, and the two flows are presented to the FlexE shim. An 800GBASE-R PHY can carry eight 100G FlexE Instances.

5.2.2.2 Collection from the PHY functions, deinterleaving and de-padding

For the case of 200G or 400G PHYs, the incoming stream from the lower layers of the PHY is 66b block deinterleaved and unpadding into two or four 100G FlexE Instances, respectively, as described in clause 6.2.

For the case of 800G FlexE PHYs, the FlexE Demux includes the processes of collecting 257b blocks from the two 400G flows and 256B/257B trans-decoding as specified in [802.3df]. Each 257b block is trans-decoded to four 66B blocks belonging to the same 100G FlexE Instance. Note that 100G FlexE instances 0, 2, 4, and 6 blocks are collected from 400G flow 0 and 100G FlexE instances 1, 3, 5, and 7 blocks are collected from 400G flow 1. The deinterleaving and de-padding of the eight 100G FlexE Instances is done in groups of four 66B blocks, as described in clause 6.2.

5.2.2.3 Calendar Interleaving and Overhead Extraction

The calendar slots of the sub-calendars on each FlexE Instance are logically interleaved in the order specified in clause 6.6. The FlexE overhead is recovered from each FlexE Instance.

5.2.2.4 LF Generator

In the case that any PHY of the FlexE Group has failed (PCS_Status=FALSE) or overhead frame lock or overhead multiframe lock has not been achieved on the overhead of any of the FlexE Instances, LF is generated towards all FlexE Clients in the group.

5.2.2.5 66B Block Extraction from Calendar

The 66B blocks are extracted from the calendar positions assigned to each FlexE Client in the order described in clause 6.5.

Note – In the case where a FlexE client extracted from this FlexE group may be forwarded to another FlexE group or to an Ethernet PHY without recoding of the 66B block stream, it is advisable to replace any block with a corrupted sync header (which may have resulted from error marking by a FEC decoder on one of the FlexE PHYs) with an error control block (see Figure 35) prior to forwarding.

5.2.2.6 Idle Insertion/Deletion

Idle insertion/deletion according to [802.3] clause 82.2.3.6 and/or ordered set deletion according to [802.3] clause 82.2.3.9 may be performed to rate-adapt the extracted 66B flow when necessary to adjust to the FlexE Client rate. Note that the nominal rate of the 66B flow carried in the calendar slots is slightly less than the nominal rate of the FlexE Client as the available space is reduced by the FlexE PHY PCS lane alignment markers the FlexE overhead, and 66B pad blocks where applicable.

5.2.2.7 Control

The control function manages which calendar slots each FlexE Client is extracted from each FlexE Instance in the receive direction.

5.3 Sample Applications

FlexE can support a variety of applications. A non-exhaustive list includes:

- Router to Transport Connection (more examples below).
- Intra-Data Center “Fat Pipe” application: bonded PHYs for flows exceeding the PHY rate or carrying traffic that doesn’t distribute efficiently with LAG.
- Generalized MLG applications, e.g., an $n \times 100\text{G}$ PHY as an umbilicus to a satellite shelf of lower rate ports.

One case of router to transport connection is where the transport network is unaware of FlexE. This case is illustrated in Figure 12. This may be used with transport equipment that provides PCS-codeword transparent transport of 50GBASE-R, 100GBASE-R, 200GASE-R, 400GBASE-R, or 800GBASE-R but provides no special support for FlexE.

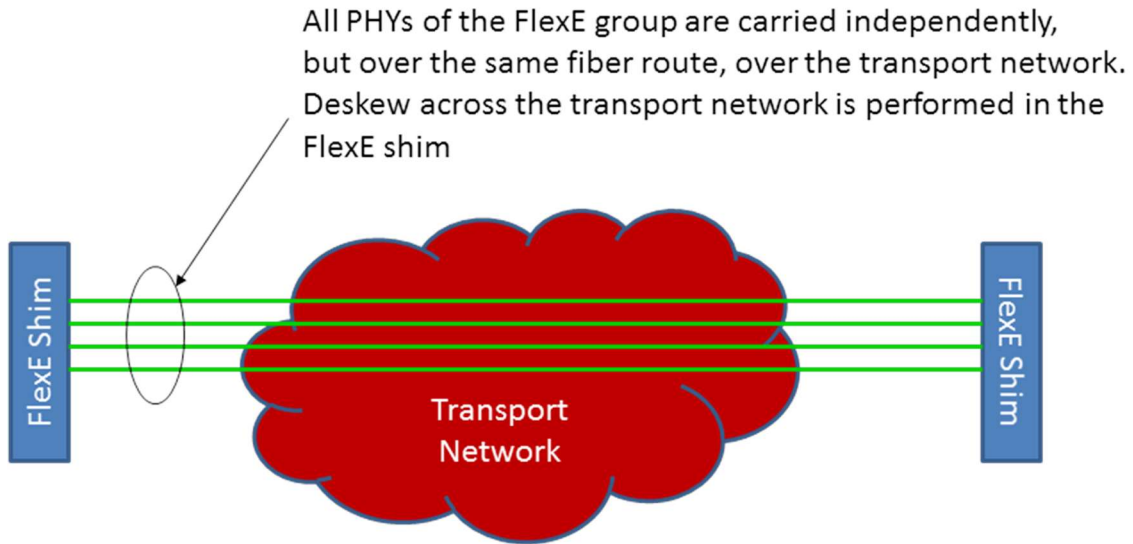


Figure 12: Router to FlexE unaware Transport Network connection

In the FlexE unaware case, the FlexE Shim, e.g., in a router, maps the FlexE Client(s) over a group of bonded Ethernet PHYs. Each of the Ethernet PHYs is carried independently over the transport network using a PCS codeword transparent mapping. The Ethernet PHYs are intended to be carried over the same fiber route: diverse routing is not envisioned. All of the PHYs of the FlexE Group need to be interconnected between the same two FlexE Shims. The FlexE Shim will need to tolerate and accommodate considerably more skew than if the FlexE Shims were only separated by an Ethernet link distance of 40km or less, as the transport network could carry the signal over thousands of kilometers. In Figure 12, it is the PHYs of the FlexE Group which are carried over the transport network.

Another case of router to transport connection is where the transport network equipment terminates the FlexE Group. This case is illustrated in Figure 13.

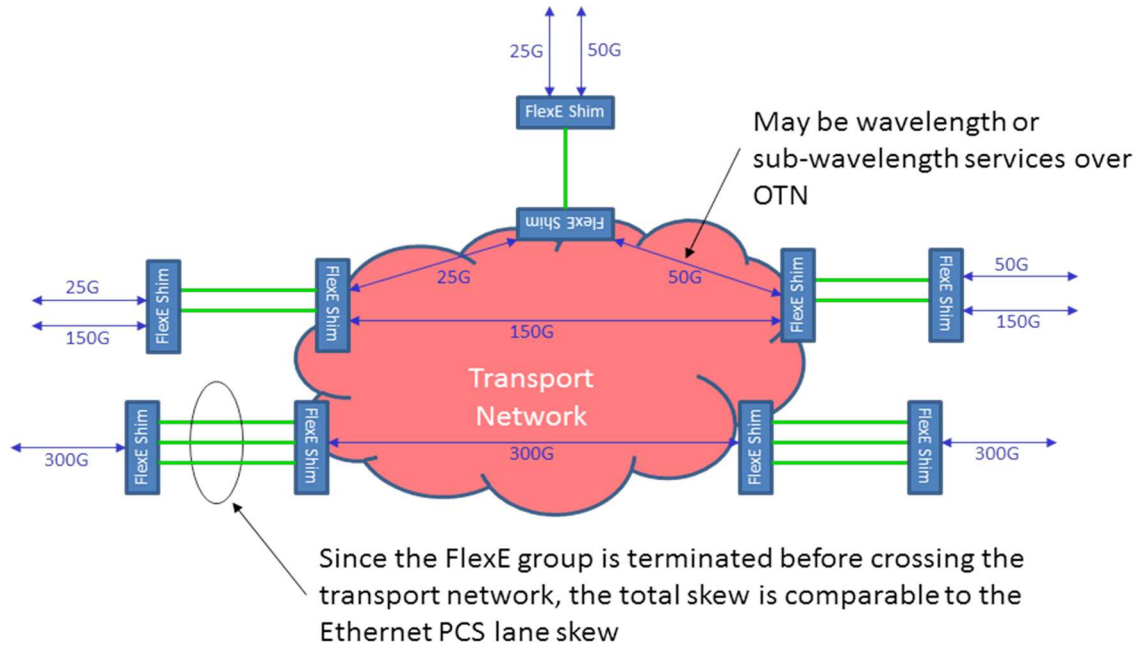


Figure 13: FlexE terminating transport network equipment

In the FlexE terminating case, the distance between any pair of FlexE Shims is limited to the Ethernet link distance (about 40km maximum), so the amount of skew that needs to be tolerated and compensated is considerably less. The other important distinction here is that it is the FlexE Clients, rather than the PHYs of the FlexE Group, which are carried over the transport network. The FlexE Client could be constructed to be the complete size of the payload that can be carried over a single wavelength (e.g., construct 200G to fill a DP-16QAM wavelength with the bonding of two 100GBASE-R PHYs), or could be a smaller client which is multiplexed and switched at a sub-wavelength level.

The final router to transport example described is one where the transport network is aware that it is carrying FlexE PHYs (as opposed to 100GbE), but the FlexE Group is not terminated on the transport equipment. This may be used to support cases where the Ethernet PHY rate is greater than the wavelength rate, the wavelength rate is not an integral multiple of the PHY rate, or there is a reason (for example, wavelengths terminated on different transponder line cards) that it is not possible to terminate the FlexE Group in the transport equipment. This kind of example is illustrated in Figure 14.

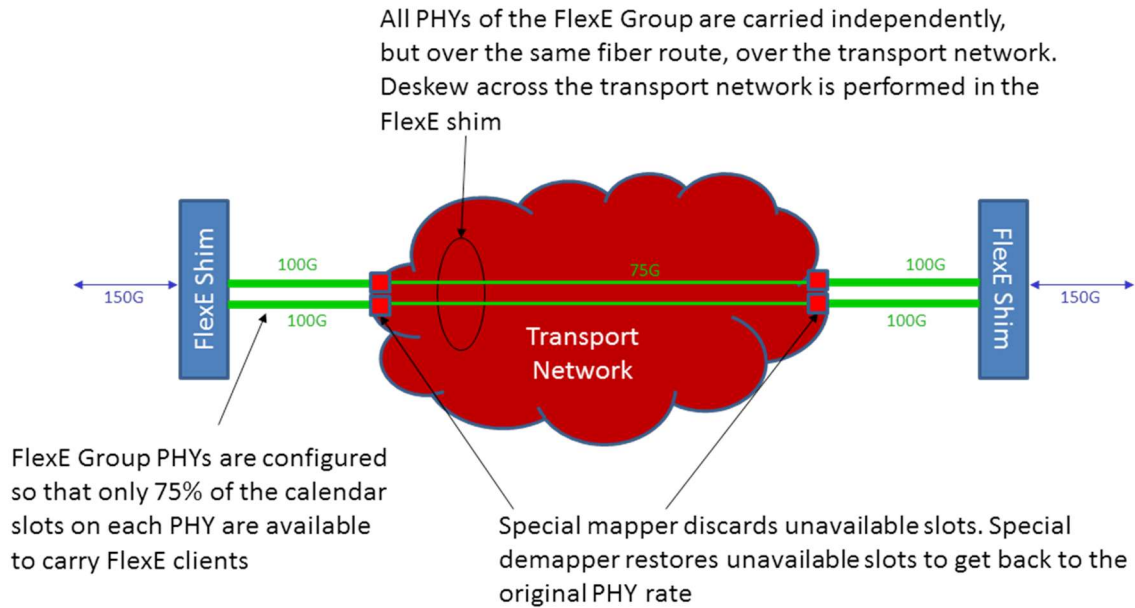


Figure 14: Example of FlexE aware transport of Ethernet PHYs of a FlexE Group

6 General Mechanism

6.1 FlexE Group

A FlexE Group is composed either of:

- from 1 to n 50G FlexE Instances which are carried over from 1 to m 50GBASE-R Ethernet PHYs; or
- from 1 to n 100G FlexE Instances which are carried over from 1 to m 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R Ethernet PHYs.

All PHYs in a group must operate at the same rate. Note that for groups composed of 50GBASE-R or 100GBASE-R PHYs, $m=n$ as there is a single FlexE Instance carried over each PHY.

- For a group composed of 50GBASE-R PHYs, each PHY is identified by a number in the range [1-126]. The values of 0 and 127 are reserved. The FlexE PHY number and the 50G FlexE Instance number for 50GBASE-R PHYs are the same. Note that because of the way 50G FlexE Instance numbers are chosen, the top bit of the FlexE Instance number field will always be zero for a 50G FlexE Instance.
- For a group composed of 100GBASE-R PHYs, each PHY is identified by a number in the range [1-254]. The values of 0 and 255 are reserved. The FlexE PHY number and the 100G FlexE Instance number for 100GBASE-R PHYs are the same.
- For a group composed of 200GBASE-R PHYs, each PHY is identified by a number in the range [1-126]. The values 0 and 127 are reserved. The two 100G FlexE Instances that may be carried by a 200GBASE-R PHY are identified by an eight-bit 100G FlexE Instance number that consists of the PHY number in the upper seven bits, and 0 or 1 in the lower order bit.

- For a group composed of 400GBASE-R PHYs, each PHY is identified by a number in the range [1-62]. The values 0 and 63 are reserved. The four 100G FlexE Instances that may be carried by a 400GBASE-R PHY are identified by an eight-bit 100G FlexE Instance number that consists of the PHY number in the upper six bits, and 0, 1, 2, or 3 in the two lower-order bits.
- For a group composed of 800GBASE-R PHYs, each PHY is identified by a number in the range [1-30]. The values 0 and 31 are reserved. The eight 100G FlexE Instances that may be carried by an 800GBASE-R PHY are identified by an eight-bit 100G FlexE Instance number that consists of the PHY number in the upper five bits, and 0-7 in the lower three bits.

A PHY number may correspond to the physical port ordering on equipment, but the FlexE Shim at each end of the group must identify each PHY in the group using the same PHY number, and each FlexE Instance with the same FlexE Instance number. PHY numbers within the group do not need to be contiguous, but 100G FlexE Instance numbers within the same 200G, 400G, or 800G PHY must be contiguous.

While the protocol allows between 30 and 254 different PHY numbers to be assigned (depending on the signaling rate), practical implementations are likely limited to the range of 4-8 PHYs.

6.1.1 Groups composed of 50GBASE-R PHYs

Each 50GBASE-R PHY uses the bulk of the PCS functions described in [802.3] clause 133 including PCS lane distribution, lane marker insertion, alignment, and deskew. All PHYs of the FlexE Group must use the same physical layer clock. Each PHY of the FlexE Group is able to deliver a logically serial stream of 64B/66B encoded blocks from the FlexE mux to the FlexE demux at a data rate of:

$$51.5625 \text{ Gb/s} \times \frac{20479}{20480} \pm 100\text{ppm}$$

The fraction applied to the base rate reflects the fact that 1/20K of the space of the interface is occupied by PCS lane alignment markers which are not space available to carry FlexE blocks. The FlexE blocks carried over each PHY of the FlexE Group has the format of a logically serial stream of legal 64B/66B blocks with the format described in [802.3] Figure 82-5 (through reference from [802.3] clause 133)), although the blocks do not appear in a sequence that would make sense if interpreted as an Ethernet interface. The actual PHYs of the FlexE Group transcode these blocks to 256B/257B format according to [802.3] clause 91.5.2.5 (through reference from [802.3] clause 134.5.2.5)), but they are trans-decoded back to 64B/66B blocks prior to delivery to the FlexE demux.

6.1.2 Groups composed of 100GBASE-R PHYs

Each 100GBASE-R PHY uses the bulk of the PCS functions described in [802.3] clause 82 including PCS lane distribution, lane marker insertion, alignment, and deskew. All PHYs of the FlexE Group must use the same physical layer clock. Each PHY of the FlexE Group is able to deliver a logically serial stream of 64B/66B encoded blocks from the FlexE mux to the FlexE demux at a data rate of:

$$103.125 \text{ Gb/s} \times \frac{16383}{16384} \pm 100\text{ppm}$$

While the protocol supports a number of PHYs in the FlexE Group up to 254, practical implementations are likely limited to the range of 4-8 PHYs. The fraction applied to the base rate reflects the fact that 1/16K of the space of the interface is occupied by PCS lane alignment markers which are not space available to carry FlexE blocks. The FlexE blocks carried over each PHY of the FlexE Group have the

format of a logically serial stream of legal 64B/66B blocks with the format described in [802.3] Figure 82-5, although the blocks do not appear in a sequence that would make sense if interpreted as an Ethernet interface. The actual PHYs of the FlexE Group may transcode these blocks to 256B/257B format according to [802.3] clause 91.5.2.5 according to the PHY type, but they are trans-decoded back to 64B/66B blocks prior to delivery to the FlexE demux.

6.1.3 Groups composed of 200GBASE-R or 400GBASE-R PHYs

Each 200GBASE-R or 400GBASE-R PHY uses the bulk of the PCS functions described in [802.3] clause 119, including 257B transcoding, scrambling, alignment insertion, pre-FEC distribution, FEC encoding, and PCS lane distribution and interleaving.

Each 200GBASE-R PHY is able to deliver a logically serial stream of 66B encoded blocks from the FlexE mux to the FlexE demux at a data rate of:

$$206.25 \text{ Gb/s} \times \frac{20479}{20480} \pm 100\text{ppm}$$

Each 400GBASE-R PHY is able to deliver a logically serial stream of 66B encoded blocks from the FlexE mux to the FlexE demux at a data rate of:

$$412.5 \text{ Gb/s} \times \frac{20479}{20480} \pm 100\text{ppm}$$

The fraction applied to the base rate reflects that 1/20K of the space of the interface is occupied by PCS lane alignment markers which are not space available to carry FlexE blocks. The FlexE blocks carried over each PHY of the FlexE Group have the format of a logically serial stream of (mostly) legal 64B/66B blocks with the format described in [802.3] Figure 82-5 (through reference from [802.3] clause 119)), although the blocks do not appear in a sequence that would make sense if interpreted as an Ethernet interface. The actual PHYs of the FlexE Group will transcode these blocks to 256B/257B, scramble, and FEC encode for transmission over the physical interface, but they are trans-decoded back to 64B/66B blocks prior to delivery to the FlexE demux.

6.1.4 Groups composed of 800GBASE-R PHYs

Each 800GBASE-R PHY uses the bulk of the PCS functions described in [802.3df] clause 172, including scrambling, alignment insertion, pre-FEC distribution, FEC encoding, and PCS lane distribution and interleaving. The functions of 66B block distribution to two flows and transcoding to 257B blocks are pulled into the FlexE shim to ensure each 257B block will contain four 66B blocks belonging to the same FlexE Instance.

Each 800GBASE-R PHY is able to deliver a logically serial stream of 66B encoded blocks from the FlexE mux to the FlexE demux at a data rate of:

$$825 \text{ Gb/s} \times \frac{20479}{20480} \pm 50\text{ppm}$$

The fraction applied to the base rate reflects that 1/20K of the space of the interface is occupied by PCS lane alignment markers which are not space available to carry FlexE blocks. The FlexE blocks carried over each PHY of the FlexE Group have the format of a logically serial stream of (mostly) legal 64B/66B blocks with the format described in [802.3] Figure 82-5 (through reference from [802.3df] clause 172)), although the blocks do not appear in a sequence that would make sense if interpreted as an Ethernet interface. The 800G FlexE mux will transcode these blocks to 256B/257B per clause 172.2.4.4 of

[802.3df], and the PHYs will scramble and FEC encode for transmission over the physical interface. The 257B blocks are delivered to the 800G FlexE demux, which will be transcoded back to 66B blocks.

6.2 FlexE Instances, padding and interleaving

Each 50GBASE-R PHY carries a single 50G FlexE Instance and one set of Pads. Pad blocks are used to compensate the difference between the 1/16K alignment marker spacing for 100GBASE-R PHYs and the 1/20K alignment marker spacing for 50GBASE-R, so that the 50G FlexE Instance bit rate is exactly half the nominal bit rate of the 100G FlexE Instance. Two 66B pad blocks (P1, P2) are inserted per 163830 payload blocks on each 50G FlexE Instance.

Each 100GBASE-R PHY carries a single 100G FlexE Instance in the 66B block positions between alignment markers.

Each 200GBASE-R PHY, 400GBASE-R PHY, or 800GBASE-R PHY carries two, four, or eight 100G FlexE Instances and two, four, or eight sets of Pads, respectively. Pad blocks are used to compensate the difference between the 1/16K alignment marker spacing for 100GBASE-R PHYs and the 1/20K alignment marker spacing for 200GBASE-R, 400GBASE-R, and 800GBASE-R PHYs so that the 100G FlexE Instances carried over all PHY types are the same nominal size. For 200GBASE-R and 400GBASE-R PHYs, two 66B pad blocks (P1, P2) are inserted per 163830 payload blocks at the same relative position on each 100G FlexE Instance. For 800GBASE-R PHYs, four pad blocks (P1, P1, P2, P2) are inserted per 327660 payload blocks at the same relative position on each 100G FlexE Instance.

- The 100G FlexE Instances carried by a 200GBASE-R, 400GBASE-R, or 800GBASE-R PHY are all aligned using the start of the FlexE overhead frame and multiframe. The Instances are then interleaved to fill the logical 66B block space between alignment markers on each respective PHY type (note that physically, the 66B blocks are 257B transcoded on these PHYs). For 200GBASE-R and 400GBASE-R PHYs, 66B blocks are interleaved. For 800GBASE-R PHYs, groups of four 66B blocks are interleaved. On a 200GBASE-R PHY, the first overhead block of the overhead multiframe for 100G FlexE Instance xxxx_xxx0 is immediately followed by the first overhead block of the overhead multiframe for 100G FlexE Instance xxxx_xxx1. This results in a sequence of two consecutive FlexE overhead blocks followed by $2 \times 20 \times 1023$ non-pad payload blocks followed by two consecutive FlexE overhead blocks, etc.
- On a 400GBASE-R PHY, the first overhead block of the overhead multiframe for 100G FlexE Instance xxxx_xx00 is immediately followed by the first overhead blocks of the overhead multiframe for 100G FlexE Instances xxxx_xx01, xxxx_xx10, and xxxx_xx11 in sequence. This results in a sequence of four consecutive FlexE overhead blocks followed by $4 \times 20 \times 1023$ non-pad payload blocks followed by four consecutive FlexE overhead blocks, etc.
- On an 800GBASE-R PHY, it is convenient to describe the pattern of blocks after interleaving in terms of four rows of the FlexE frame, each row consisting of one overhead block and 20460 payload blocks, for a total of 81844 blocks. The blocks of each 100G FlexE Instance are interleaved in groups of four. The first overhead block of the overhead multiframe for 100G FlexE Instance xxxx_x000 is immediately followed by 3 data blocks from that Instance, and then the first overhead block of the overhead multiframe and three data blocks of each of the other seven 100G FlexE Instances in sequence. This is followed by $8 \times (20 \times 1023 - 4)$ non-pad payload blocks, and then by 32 blocks that are overhead plus 3 payload blocks of each Instance, etc. Since the number of blocks in a row is not divisible by four, the position of the overhead block

within the set of 4 blocks that are interleaved will shift, producing a pattern of blocks that repeats every $4 \times 8 \times (20 \times 1023 + 1) = 654,752$ blocks.

The alignment of the FlexE Overhead blocks of the interleaved 100G FlexE Instances on 200GBASE-R, 400GBASE-R, and 800GBASE-R PHYs, excluding the pad blocks is illustrated in Figure 15. For 800G, only the first of the four frames, plus the start of the second one, is shown. Notice that the position of the second set of OH blocks (at the right side of the figure) has moved within the 4-block interleaving unit compared to where it was in the first set of OH blocks (at the left side of the figure).

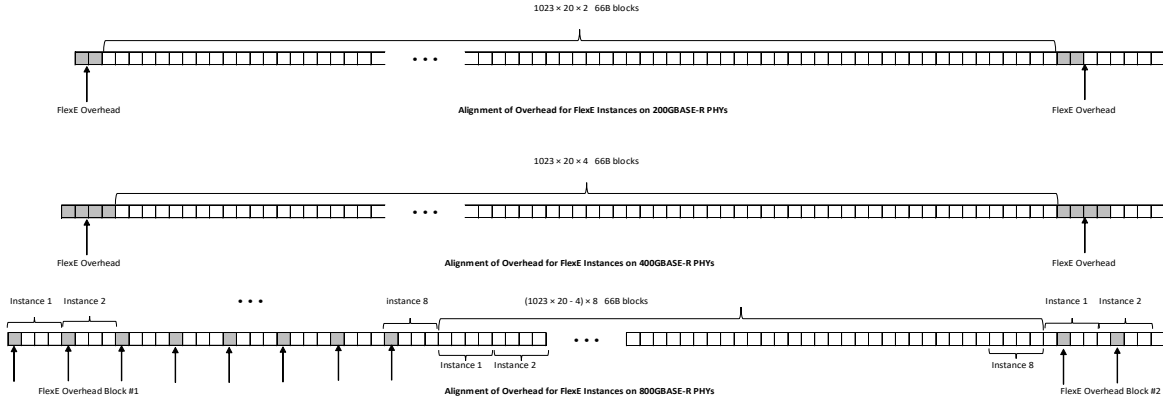


Figure 15: Alignment of Overhead on Interleaved 100G FlexE Instances on 200GBASE-R, 400GBASE-R, and 800GBASE-R PHYs

Since the pad blocks are inserted at the same relative position on each 100G FlexE Instance, a consequence of this alignment is that the pad blocks will occur on 200GBASE-R PHYs in groups of four pad blocks followed by 2×163830 payload blocks followed by four pad blocks, etc., and on 400GBASE-R PHYs in groups of eight pad blocks followed by 4×163830 payload blocks followed by eight pad blocks, etc. On 800GBASE-R PHYs, groups of 32 pad blocks will be followed by 8×327660 payload blocks. While this implementation agreement is written in terms of pad block insertion and removal on a 100G FlexE Instance basis, implementations are free to insert or remove pad blocks on a PHY basis (or, in the case of 800G, a per-flow basis), or in a combined state machine with FlexE overhead insertion and removal, as long as the position of the pad blocks on the PHY is correct. Figure 16 illustrates the distribution of pad blocks on 50GBASE-R, 200GBASE-R, 400GBASE R in terms of 66B equivalent blocks between alignment markers.

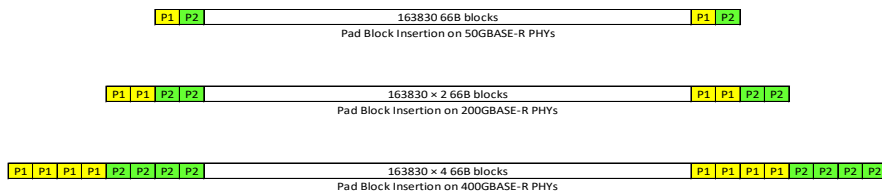


Figure 16: Distribution of Pad blocks on 50GBASE-R, 200GBASE-R and 400GBASE-R PHYs

Figure 17 illustrates the distribution of pad blocks on 800GBASE-R in terms of 66B equivalent blocks. Since the insertion of pad blocks is only half as often on 800GBASE-R PHYs, the relationship to alignment markers will be different than it is for 50GBASE-R, 200GBASE-R, and 400GBASE-R PHYs.

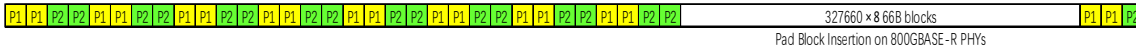


Figure 17: Distribution of Pad blocks on 800GBASE-R PHYs

The format of the P1 pad block is indicated in Figure 18. This uses the same ordered set “O” code as is used to identify the FlexE overhead, with the value 0xFFFF in bits 12-31 (where the FlexE Group number would be for the first block of the FlexE overhead frame) distinguishing this as a pad block rather than a FlexE overhead block. Bits 8-11 of the P1 pad block are set to zero.

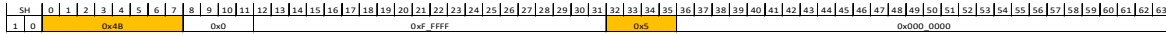


Figure 18: Format of P1 Pad Block

The format of the P2 pad block is an Ethernet error control block as shown in Figure 35.

6.3 Unequipped 100G FlexE Instances

For PHYs with more than one 100G FlexE Instance, any 100G FlexE Instance not intended to be used to carry FlexE client traffic may be configured as unequipped, which avoids the necessity to process the overhead and manage the calendar for that 100G FlexE Instance. This could be used, for example, in a FlexE aware configuration with 300G of transport network capacity with a single-member 400GBASE-R FlexE Group. As described in clause 8.3, unequipped FlexE instances are discarded at the transport network ingress node and reinserted at the transport network egress node.

The following rules govern unequipped instances:

- Unequipped instances must always be the highest numbered instance(s) on a PHY of the FlexE Group.
- The first instance in a PHY cannot be configured as unequipped because it carries overhead for the PHY (such as the management channels and RPF).

The format of an unequipped 100G FlexE Instance is as follows:

- The first block of the overhead frame is as shown in Figure 19. This is essentially the same as in any other overhead frame, except that the FlexE Group number is indicated as 0x00000, a reserved value to indicate that this instance doesn’t belong to the group. Bits 8-11 of this block are set to zero.

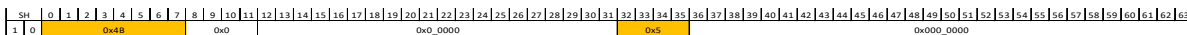


Figure 19: Format of First Block of FlexE overhead frame for unequipped 100G FlexE Instances

- All other non-pad blocks of the unequipped 100G FlexE Instance (including the remaining blocks of the overhead frame) are filled with Ethernet error control blocks as indicated in Figure 35. This ensures that no blocks from an unequipped 100G FlexE Instance can be accidentally interpreted as valid Ethernet data in the case of misconfiguration.
- Pad blocks are inserted on unequipped 100G FlexE Instances in the same relative position as for equipped instances as described in 6.2.

To avoid the need to process the calendar overhead, unequipped 100G FlexE Instances are not indicated as being part of the FlexE map in describing the composition of the FlexE Group (see 7.3.3).

6.4 FlexE Client rate adaptation

The FlexE Client presented to the FlexE Shim is in the format described in clause 5.2.1.1.

All FlexE Clients to be transmitted over the same FlexE group are aligned to a common clock and rate-adapted to the available space in the FlexE calendar (whose nominal rate is slightly less than that of the FlexE Client due to space needed for the FlexE PHY PCS lane alignment markers and FlexE overhead) according to the process described in clause 5.2.1.2. The rate-adapted FlexE Client operates at a rate of:

$$\text{FlexE Client MAC rate} \times \frac{66}{64} \times \frac{16383}{16384} \times \frac{1023 \times 20}{1023 \times 20 + 1} \pm 100\text{ppm or } 50\text{ppm}$$

This nominal rate is about 0.011% less than the nominal rate of the FlexE Client, which is well within what can be accomplished with idle insertion/deletion without packet loss. Note that this doesn't actually correspond to any clock that needs to be generated in an implementation, as the idle insertion and deletion process will simply operate by filling the allocated block positions in the FlexE Group from a FlexE Client FIFO, inserting or deleting idles in the process of filling the block positions in the FlexE Group according to the calendar (see below).

6.5 FlexE Calendar

The FlexE mechanism operates using a calendar which assigns 66B block positions on sub-calendars on each FlexE Instance of the FlexE Group to each of the FlexE Clients. The calendar is described with a granularity of 5G (although implementations may limit bandwidth assignment to coarser granularity, e.g., 25G or 100G), and has a length of ten 5G slots for each 50G FlexE Instance or twenty 5G slots for each 100G FlexE Instance in the group. Two calendar configurations are supported: an "A" and a "B" calendar configuration. At any given time, one of the calendar configurations is used for mapping the FlexE Clients into the FlexE Group and demapping the FlexE Clients from the FlexE Group. The two calendar configurations are provided to facilitate reconfiguration. When a switch of calendar configurations adds or removes FlexE Clients from the FlexE Group, existing clients whose size and calendar slot assignments are not changed by changing the calendar configuration are not affected.

For a FlexE Group composed of n 50G FlexE Instances, the logical length of the calendar is 10n. The blocks as allocated per the calendar are distributed to n sub-calendars of length 10 on each of the 50G FlexE Instances of the FlexE Group according to Figure 20.

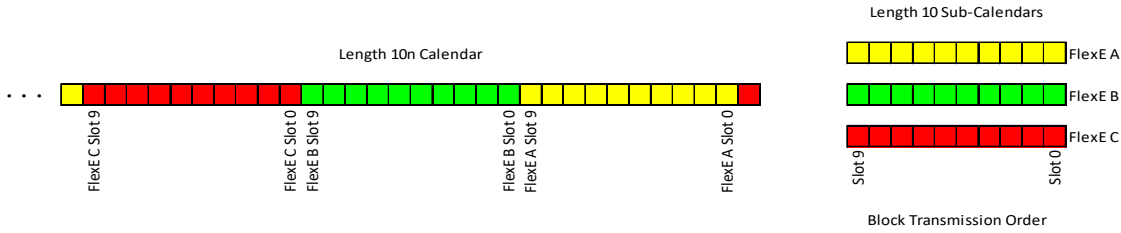


Figure 20: Illustration of FlexE Calendar Distribution based on 5G granularity to length 10 sub-calendars (50G FlexE Instances)

The order of distribution of ten blocks at a time is selected over simple “round robin” distribution of 66B blocks. Calendar slots are identified by their 50G FlexE Instance number and the slot [0-9] (within that 50G FlexE Instance). The “logical” sequence number of a calendar slot is 10×the instance number plus the calendar slot number within the 50G FlexE Instance. The sequence is ascending order. Note that the sequence numbering is not necessarily consecutive when the assigned PHY numbers are not contiguous. This logical order only matters when calendar slots on different 50G FlexE Instances are assigned to the same FlexE Client.

For a FlexE Group composed of n 100G FlexE Instances, the logical length of the calendar is 20n. The blocks as allocated per the calendar are distributed to n sub-calendars of length 20 on each of the 100G FlexE Instances of the FlexE Group according to Figure 21.

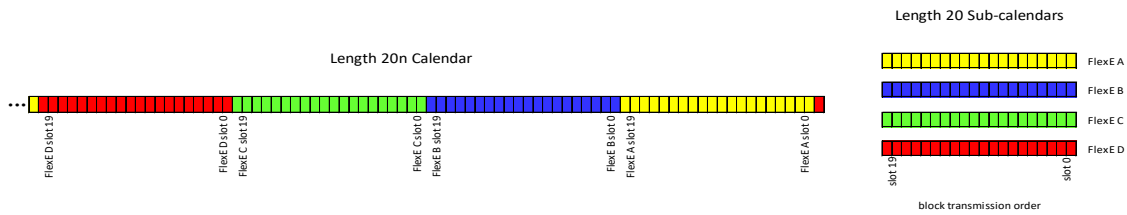


Figure 21: Illustration of FlexE Calendar Distribution based on 5G granularity (100G FlexE Instances)

The order of distribution of twenty blocks at a time is selected over simple “round robin” distribution of 66B blocks. Calendar slots are identified by their 100G FlexE Instance number and the slot [0-19] (within that 100G FlexE Instance). The “logical” sequence number of a calendar slot is 20 times the instance number plus the calendar slot number within the 100G FlexE Instance. The sequence is ascending order. Note that the sequence numbering is not necessarily consecutive when the assigned PHY numbers are not contiguous. This logical order only matters when calendar slots on different 100G FlexE Instances are assigned to the same FlexE Client.

Implementations may limit the bandwidth assignment granularity to 25G rather than 5G as illustrated in Figure 22. An effective 25G calendar slot occupies the same space as five consecutive 5G calendar slots: for example, slots 0-4, 5-9, 10-14 and 15-19 in a 100G FlexE Instance are combined to create a calendar that is logically length 4 for a 100G FlexE Instance. The FlexE Client assignments in a 25G granularity calendar are indicated by constraining that groups of five consecutive 5G calendar slots must always carry the same FlexE Client.

Maintaining the description of the calendar at 5G granularity permits interoperability between implementations supporting 5G calendar slots and implementations supporting 25G calendar slots by requiring the 5G-capable implementation to always assign the same client to groups of five consecutive slots when connected to a 25G capable implementation.

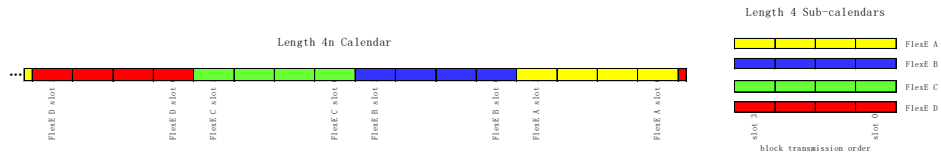


Figure 22: Illustration of FlexE calendar distribution based on 25G granularity

Implementations may limit the bandwidth assignment granularity to 100G rather than 5G or 25G, as illustrated in Figure 23. An effective 100G calendar slot occupies the same space as all twenty 5G or four 25G calendar slots in a 100G FlexE instance.

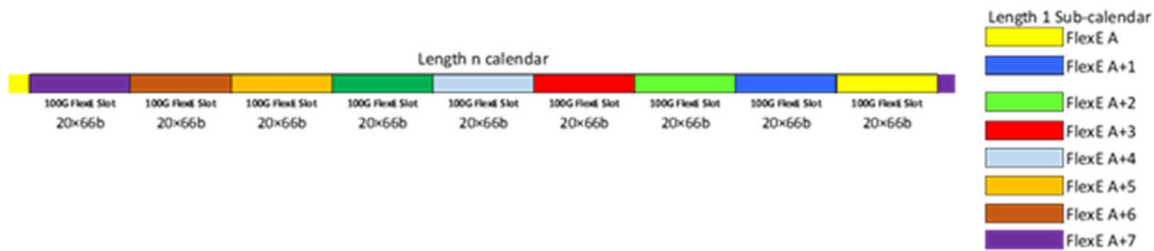


Figure 23: Illustration of FlexE calendar distribution based on 100G granularity

Maintaining the description of the calendar at 5G granularity permits interoperability between implementations supporting 5G or 25G calendar slots and implementations supporting 100G calendar slots by requiring the 5G-capable implementation to always assign the same client to groups of twenty consecutive slots when connected to a 100G capable implementation.

6.6 FlexE Overhead and Alignment

The alignment of the data from the FlexE Instances of the FlexE Group is accomplished by the insertion of FlexE overhead into the stream of 66B blocks carried over the group. The FlexE overhead is delineated by a 66B block which can be recognized independently of the FlexE Client data. An illustration of the FlexE overhead on each FlexE Instance of the FlexE Group is given in Figure 24 or Figure 26.

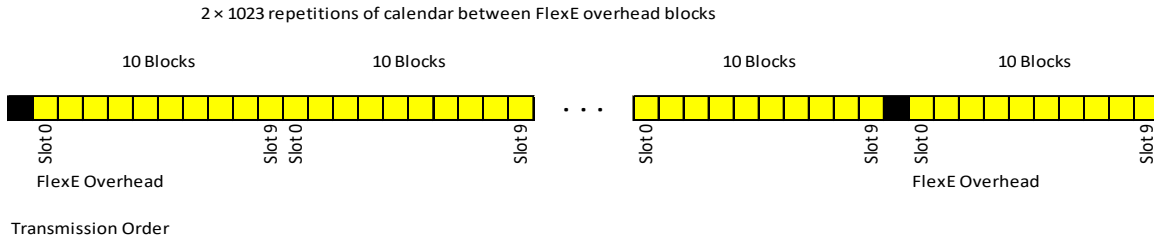


Figure 24 : Illustration of insertion of FlexE overhead on each 50G FlexE Instance of a FlexE Group (50GBASE-R PHYs only)

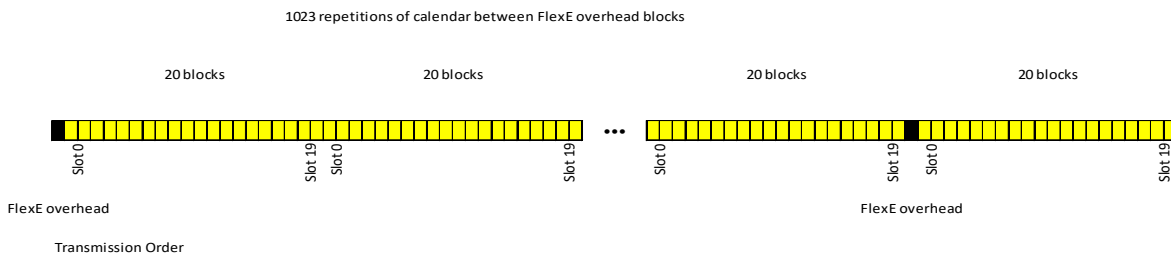


Figure 25: Illustration of insertion of FlexE overhead on each 100G FlexE Instance of a FlexE Group (100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R PHYs)

On a 50G FlexE Instance, a FlexE overhead block will occur approximately once per 26.2μs. On a 100G FlexE Instance, a FlexE overhead block will occur approximately once per 13.1μs. The actual format of the FlexE overhead blocks is such that they occur in a repeating sequence of eight blocks, so the sequence has a period of approximately 209.52μs for 50G FlexE Instance, or 104.77μs for a 100G FlexE Instance. This sequence of overhead blocks is inserted in the same positions in the sub-calendar sequence on each FlexE Instance and is used to align all of the FlexE Instances of the FlexE Group at the FlexE demux to reconstruct the sequence in the order of the calendar so that the FlexE Clients can be recovered.

The FlexE aware transport network scenario illustrated in Figure 14 allows for marking a certain number of the calendar slots as unavailable. This is different from “unused”, in that it is known, due to transport network constraints, that not all of the calendar slots generated from the FlexE mux will reach the FlexE demux and therefore no FlexE Client should be assigned to those slots. The intention is that when a 100G FlexE Instance of the FlexE Group is carried across the transport network, the mapping is able to compress the signal to less than the 100G FlexE Instance rate by dropping the unavailable calendar slots. A case where 25% of the calendar slots are unavailable is illustrated in Figure 27. Unavailable slots are placed at the end of each relevant sub-calendar (the highest numbered slots).

Note that the OTN mapping defined in [G.709] to support FlexE Aware Transport (see clause 8.3) requires that every equipped 100G FlexE instance have at least one available calendar slot. This implies that:

- Any PHY without at least one available calendar slot should not be part of the FlexE group.

100G FlexE Instance on a 200GBASE-R, 400GBASE-R, or 800GBASE-R PHY) is available for communication across a section (for example, from a router with a FlexE Shim to FlexE aware transport equipment which does not terminate the FlexE Shim), and the other management channel (the 6th-8th or 7th-8th blocks of the FlexE overhead frame of the first FlexE Instance on a given PHY) is used for communication between the FlexE Shims.

- An optional synchronization messaging channel. If configured, this is carried in the 6th block of the FlexE overhead frame of the first FlexE Instance on a given PHY.
- Payload Type codepoints for FlexE 3.0 implementations or later to indicate the composition of the FlexE Group payload (i.e., calendar slot granularity, test pattern, protocol versions, etc.).

The amount of information to be conveyed from the FlexE mux to the FlexE demux exceeds the 24 bits available in a single ordered set block per FlexE Instance. This is addressed by spreading the relevant overhead across a sequence of eight FlexE overhead blocks on each FlexE Instance, each separated by 20×1023 FlexE data blocks. This group of eight overhead blocks is referred to as the FlexE overhead frame. The FlexE overhead frame is illustrated in Figure 29 for a 50G FlexE Instance and Figure 30 for a 100G FlexE Instance. The meaning, interpretation and processing of this overhead is explained in clause 7.3.

7 Detailed Functions

Some detailed processing to implement the functionality described in clause 6 is provided in this clause.

Implementations may use the processes and functions of equipment functional blocks supporting Flex Ethernet interfaces specified in ITU-T [G.8023].

7.1 FlexE Group Functions

The FlexE Group is composed of either:

- from 1 to n 50G FlexE Instances which are carried over from 1 to m 50GBASE-R PHYs; or
- from 1 to n 100G FlexE Instances which are carried over from 1 to m 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R PHYs.

Note that for groups composed of 50GBASE-R or 100GBASE-R PHYs, $m=n$ as there is a single FlexE Instance carried over each PHY.

The FlexE Shim provides to each FlexE Group PHY a set of 64B/66B encoded blocks that are encoded according to Figure 82-5.

7.2 FlexE Client Generation

The format and bit rate of FlexE Clients is described in clause 6.4. FlexE Clients generally originate from one of the following sources.

7.2.1 FlexE Clients Generated internally within a system

A FlexE Client may be generated internally within a system, for example from a Network Processing Unit (NPU) within a router. The packet flow is generated at the determined FlexE Client MAC rate and 64B/66B encoded according to [802.3] Figure 82-5.

7.2.2 FlexE Clients received from an Ethernet PHY

FlexE Clients at the rates of 10G, 25G, 40G, 50G, 100G, 200G, 400G, and 800G (per existing [802.3] and [802.3df] standards) can be created from an Ethernet PHY at the corresponding rate with some processing to convert to the FlexE Client format and rate.

A 10GBASE-R signal will be converted to a 10G FlexE Client format before presenting to a FlexE mux by converting the coding from the 66B codeword set of [802.3] Figure 49-7 to the 66B codeword set of [802.3] Figure 82-5. This performs idle insertion/deletion in groups of four idles and or ordered set deletion where necessary to align the start control character to an 8-byte boundary (eliminating the use of control block types 0x2d, 0x66, and 0x33), converting control blocks that contain two ordered sets to a control block that contains one ordered set (changing control block type 0x55 to 0x4b by deleting one of the two ordered sets encoded in the block), and by replacing the unnecessary idle control characters from the end of the Figure 49-7 control block type 0x4b with zeros to produce the Figure 82-5 control block format with control block type 0x4b. A 10G FlexE Client may be converted to a 10GBASE-R signal by using the idle insertion/deletion process as described in [802.3] clause 49.2.4.7 to adapt to the 10GBASE-R nominal rate, ensuring that at least four idles appear between packets. Idles may be inserted or deleted in groups of four, and/or ordered sets may be deleted according to [802.3] clause 49.2.4.10, resulting in the start of packet on a four-byte boundary rather than the 8-byte boundary. Ordered sets encoded with control block type 0x4b have the zeros at the end of the block replaced with four idle control characters. The result is that the blocks are encoded according to Figure 49-7.

A 25GBASE-R signal will be converted to a 25G FlexE Client format before presenting to a FlexE mux by converting the coding from the 66B codeword set of [802.3] Figure 49-7 to the 66B codeword set of [802.3] Figure 82-5 in a similar manner as described for 10GBASE-R signal above. A 25G FlexE Client may be converted to a 25GBASE-R signal by using the idle insertion/deletion process as described in [802.3] clause 49.2.4.7 to adapt to the 25GBASE-R nominal rate, ensuring that at least four idles appear between packets in a similar manner as described for 10G FlexE Client signal above.

A 40GBASE-R signal can be converted to a FlexE Client by serializing and deskewing the PCS lanes, removing the PCS lane alignment markers, and using the idle insertion/deletion process as described in [802.3] clause 82.2.3.6 and/or ordered set deletion as described in [802.3] clause 82.2.3.9 to adapt the signal to the 40G FlexE Client rate. A 40G FlexE Client coming from a FlexE demux is converted to a 40GBASE-R interface by using the idle insertion/deletion process as described in [802.3] clause 82.2.3.6 and/or ordered set deletion according to [802.3] clause 82.2.3.9, distributing the blocks round-robin to the four PCS lanes, and inserting PCS lane alignment markers.

A 50GBASE-R signal will correct any errors per the FEC code, remove the FEC, trans-decode from 256B/257B, re-map from 2 FEC lanes to 4 PCS lanes, serialize and deskew the PCS lanes, and remove the PCS lane alignment markers. The idle insertion/deletion process as described in [802.3] clause 82.2.3.6 and/or ordered set deletion as described in [802.3] clause 82.2.3.9 is then used to adapt the signal to the 50G FlexE Client rate. A 50G FlexE client coming from a FlexE demux is converted to a 50GBASE-R interface by using the idle insertion/deletion process as described in [802.3] clause 82.2.3.6 and/or ordered set deletion according to [802.3] clause 82.2.3.9, distributing the blocks round-robin to the four PCS lanes, and inserting PCS lane alignment markers. The four PCS lanes are remapped to 2 FEC lanes, transcoding to 256B/257B, and adding the FEC parity as described in [802.3] clause 134.

A 100GBASE-R signal without FEC can be converted to and from a FlexE Client in the same manner as 40GBASE-R described above (except that the number of PCS lanes is 20 rather than 4). A 100GBASE-R signal with FEC, in converting to a FlexE Client, also will correct any errors per the FEC code, remove the FEC, and trans-decode from 256B/257B prior to the idle insertion/deletion process. To convert a 100G FlexE Client coming from a FlexE demux to a 100GBASE-R signal with FEC involves the same processes as for 40GBASE-R, but in addition, transcoding the signal to 256B/257B, inserting the FEC lane alignment markers, and adding the FEC.

A 200GBASE-R or 400GBASE-R signal with FEC, in converting to a 200G or 400G FlexE Client, will correct any errors per the FEC code, remove the FEC, and trans-decode from 256B/257B prior to the idle insertion/deletion process. To convert a 200G or 400G FlexE Client coming from a FlexE demux to a 200GBASE-R or 400GBASE-R signal with FEC involves the same processes as for 40GBASE-R, but in addition, transcoding the signal to 256B/257B, inserting the FEC lane alignment markers, and adding the FEC.

An 800GBASE-R signal, in converting to an 800G FlexE Client, will correct any errors per the FEC code in each of the two 400G flows, remove the FEC, trans-decode to 66B blocks, and collect the blocks from the two flows into a single stream prior to the idle insertion/deletion process. To convert an 800G FlexE Client coming from a FlexE demux to an 800GBASE-R signal involves the same processes as for 40GBASE-R, but in addition, distributing the 66B blocks to two flows prior to transcoding each flow to 256B/257B, inserting FEC lane alignment makers, and adding the FEC.

7.2.3 FlexE Clients from another FlexE Shim

In the case of equipment which terminates the FlexE Group, FlexE Clients can be delivered from the one FlexE Shim to another: for example, from a FlexE Shim at the transport network ingress to another FlexE Shim at the transport network egress. The FlexE Client, a sequence of 64B/66B encoded blocks, is expected to be carried over the transport network without packet loss. As no timing information is carried by this stream, idle insertion/deletion and ordered set deletion are possible in the mapping over the transport network. The FlexE Shim at the network egress will only need to perform idle insertion/deletion according to [802.3] clause 82.2.3.6 and/or ordered set deletion according to [802.3] clause 82.2.3.9, not due to any expected change in the nominal bit rate, but simply to align the clock with the FlexE Group clock.

7.2.4 Interconnect flexibility

Note that since the format of the FlexE Client is simply a logically serial stream of 66B blocks at a given rate, FlexE Clients do not need to be produced or received in the same manner at both ends of the connection. For example, a 10G, 40G or 100G FlexE Client might be generated as a system internal signal in the main chassis of a system, connected using an $n \times 100\text{G}$ FlexE umbilicus to a satellite shelf, and connected to physical 10GBASE-R, 25GBASE-R, 40GBASE-R, 50GBASE-R, 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R ports on the satellite shelf. In the case where the FlexE mux is receiving a FlexE Client from a physical Ethernet port and the FlexE demux is delivering that FlexE Client to a physical Ethernet port, the two ports obviously have to be the same nominal rate, but they may be different PHY types.

7.3 FlexE Overhead Processing

The format of the FlexE overhead is indicated in Figure 29 for 50G FlexE Instances and in Figure 30 for 100G FlexE Instances.

7.3.1 FlexE Overhead Frame and Multiframe Lock

The FlexE overhead is encoded as 66B blocks and are inserted on each FlexE Instance of the FlexE Group, with a sequence of one block of overhead followed by 1023×20 blocks of data followed by one block of overhead.

- For a 50G FlexE Instance, one overhead block is inserted after every 2×1023 iterations of the length 10 sub-calendar of 66B FlexE data blocks
- For a 100G FlexE Instance, one overhead block is inserted after every 1023 iterations of the length 20 sub-calendar of 66B FlexE data blocks.

FlexE overhead frame lock is achieved at the receiver (FlexE demux) on each PHY by recognizing the FlexE block 1 of the FlexE overhead frame, encoded as a special ordered set (the sync header is 10, the control block type is 0x4B (ordered set), and the "O" code is 0x5), and then finding the FlexE ordered set block again $(1023 \times 20 + 1) \times 8$ block positions later. Once FlexE overhead frame lock is achieved, the next expected FlexE overhead block will be $1023 \times 20 + 1$ block positions later. While in FlexE overhead frame lock, bytes D1-D3 of the ordered set block, plus the 66B blocks occurring at 20461, 40922, 61383, 81844, 102305, 122766, and 143227 blocks beyond the ordered set block will be interpreted as FlexE overhead frame. FlexE overhead is not interpreted if not in FlexE overhead lock. FlexE overhead lock will be lost if the sync header, control block type, or O code do not match at the expected position for 5 occurrences.

Certain information is transmitted in every FlexE overhead frame. Other information is distributed across a sequence of 16 FlexE overhead frames (for a 50G FlexE instance) or 32 FlexE overhead frames (for a 100G FlexE instance), referred to as the FlexE overhead multiframe. The OMF (overhead multiframe) bit has a value of “0” for the first eight (for a 50G FlexE instance) or sixteen (for a 100G FlexE instance) overhead frames of the overhead multiframe, and a value of “1” for the last eight (for a 50G FlexE Instance) or sixteen (for a 100G FlexE instance) overhead frames of the overhead multiframe, as shown in Figure 29 or Figure 30. The FlexE demux achieves overhead multiframe lock on each FlexE Instance when the OMF bit transitions from a “0” to a “1” or a “1” to a “0” in consecutive overhead frames with good CRC. There are two opportunities in the overhead multiframe for the FlexE demux to achieve overhead multiframe lock.

7.3.2 Calendar Configuration in Use

There are two calendar configurations for each FlexE Instance of the FlexE Group: the “A” calendar configuration (encoded as 0) and the “B” calendar configuration (encoded as one). The two calendars are used to facilitate reconfiguration. Clients can be added or removed from the FlexE Group without affecting the traffic on other clients, assuming the operating clients are carried in the same calendar slot locations. While clients can be resized or moved to different calendar slots through calendar updates, there is no assurance that the resizing or client moves can be “hitless” in all network scenarios. Normally, changes are only made to the calendar which is not currently in use.

Exceptions might include initial link configuration or replacement of a failed circuit pack where it is necessary to download the calendar information into the replacement pack. The data flow through the FlexE mux/demux pair is indeterminate during any changes to the active calendar configuration, until the assignment for all slots in the active calendar configuration have stabilized and been received by the FlexE demux in FlexE overhead frames with good CRC.

The calendar configuration in use is signaled from the FlexE mux to the FlexE demux on each FlexE Instance in the three bit positions labeled “C” in Figure 29 or Figure 30. While most of the FlexE overhead can be reliably protected by the CRC, the calendar configuration in use must be interpreted even if the CRC is bad, since the FlexE demux must be able to switch its calendar in use at precisely the same overhead frame boundary as the FlexE mux. So that this can be done reliably, three copies of the calendar configuration in use are transmitted, and interpreted by the receiver by majority vote. Since the three copies are separated into different FlexE overhead blocks across the overhead frame (the first and second copy are separated by 1,350,415 bits and the second and third copies are separated by 1,350,425 bits), the different copies will never be affected by the same burst error. Since 50GBASE-R and 100GBASE-R interfaces have a BER of 10^{-12} or better, and 200GBASE-R, 400GBASE-R, and 800GBASE-R interfaces have a BER of 10^{-13} or better, the probability of two C bits indicating the calendar in use being wrong is no more than 10^{-24} , which can safely be ignored.

When the calendar configuration in use changes from a 0 to a 1, or from a 1 to a 0, the calendar configuration used by both the FlexE mux and the FlexE demux will be changed beginning from the first data block following first block of the next FlexE overhead frame on each FlexE Instance.

7.3.3 FlexE Map and FlexE Instance Number

The set of FlexE Instances in the FlexE Group (not necessarily consecutive FlexE Instance numbers) are indicated in the “FlexE Map” field of the FlexE overhead. This is distributed as eight bits per overhead frame in each of the overhead frames in the overhead multiframe.

- For a 50G FlexE Instance, there are sixteen overhead frames in the overhead multiframe, and therefore 128 bits total;
- For a 100G FlexE Instance, there are thirty-two overhead frames in the overhead multiframe, and therefore 256 bits total.

Each bit set to a one indicates a FlexE Instance number that is a member of the group, and all other bits of the FlexE map set to zero. The FlexE Map values are only accepted from overhead frames with good CRC. The full FlexE map is sent on all FlexE Instances of the FlexE Group so that it is possible for the FlexE demux to verify that the same FlexE Instance numbers are configured at the FlexE mux as at the FlexE demux, and can tell whether all expected FlexE Instances are being received.

The FlexE Instance Number (as determined by the PHY number and the position of the instance within the PHY; see 6.1) is encoded in the second block of the FlexE overhead frame. Note that this is persistent information which does not change while the group is in service. The receiver accepts a value for “FlexE Instance Number” when the same value is received in two consecutive overhead frames with good CRC. Updates to the respective group of eight bits of the FlexE map bit map are accepted from overhead frames with good CRC.

A detailed view of the FlexE Instance Number field (see Figure 29 and Figure 30) is illustrated in Figure 31 .

FlexE Instance Number field (LSB..MSB)							
9	10	11	12	13	14	15	16
50G FlexE PHY Number							0
100G FlexE PHY Number							
0/1	200G FlexE PHY Number						
00/10/01/11		400G FlexE PHY Number					
000 - 111			800G FlexE PHY Number				

Figure 31: Structure of FlexE Instance Number field

7.3.4 Calendar Configuration

The contents of both the A and B calendar configurations are transmitted continuously from the FlexE mux to the FlexE demux, with the contents of one calendar slot of the A and B sub-calendars for each 50G FlexE Instance being transmitted in the first ten overhead frames of the FlexE overhead multiframe, and one calendar slot of the A and B sub-calendars for each 100G FlexE Instance being transmitted in the first twenty overhead frames of the FlexE overhead multiframe. The client fields are ignored by the FlexE demux when not in overhead multiframe lock since the FlexE demux would not know which slot in which calendar that client belongs to.

Note that implementations supporting 25G calendar slots signal the contents of the A/B calendars by indicating the same FlexE Client assignment in groups of the five 5G calendar slot positions that correspond to a given 25G calendar slot. Implementations supporting 100G calendar slots signal the contents of the A/B calendars by indicating the same FlexE Client assignment in groups of the twenty 5G calendar slot positions that correspond to a given 100G calendar slot.

The sub-calendar configurations on each FlexE Instance are transmitted by sending the clients assigned to each calendar slot in the same order as the corresponding 66B payload block positions occur in the transmission sequence on that FlexE Instance.

The Client fields indicate which of the FlexE Clients is mapped into a given calendar slot in the A and B calendar configurations for the sub-calendar carried over that FlexE Instance. The size of a given FlexE Client can be calculated based on the number of calendar slots that client is assigned to (i.e., how many calendar slots have the same numeric value in the Client field across the entire FlexE Group). The Clients are indicated by 16-bit fields transmitted in the 3rd block of the FlexE overhead frame. The value 0x0000 indicates a calendar slot which is unused (but available). The value 0xFFFF (all ones) indicates a calendar slot that is unavailable, for the case indicated in Figure 14 where the full FlexE Group PHY rate cannot be carried over the transport network in the FlexE aware transport use case. Any value other than 0x0000 or 0xFFFF may be used to designate a particular FlexE Client carried by the group.

The Client fields are ignored in overhead frames with a bad CRC, leaving previous assignments to the clients in the relevant slot unchanged.

The full contents of both calendar configurations are transmitted from the FlexE mux to the FlexE demux approximately once every 3.35ms. Note that while the overhead frame for a 50G FlexE Instance takes twice as long to transmit as for a 100G FlexE Instance, the overhead multiframe of a 50G FlexE Instance is half that of a 100G FlexE Instance, and therefore the time to transmit the full calendar is the same for both. The fact that the calendar configurations are transmitted continuously avoids any inconsistency between the calendars at the FlexE mux and the FlexE demux due to a lost message.

The normal process of reconfiguration (e.g., adding or removing FlexE Clients to or from the FlexE Group) will involve programming the new or modified FlexE Client assignments into the calendar configuration which is not in use, then switching to the updated calendar configuration, and finally updating the original calendar configuration.

The switch from one active calendar configuration to another can be coordinated between the FlexE mux and the FlexE demux using the Calendar Request (CR) bit sent from the FlexE mux to the FlexE demux, and the Calendar Acknowledge (CA) bit sent from the FlexE demux to the FlexE mux. Normally, the CR bit has the same value as the active calendar configuration being sent from the FlexE mux to the FlexE demux, and the CA bit has the same value as the calendar configuration currently being used by the FlexE demux.

When the FlexE mux has completed the programming of the offline calendar configuration and is ready to switch, it informs the FlexE demux by changing the CR bit to the value of the offline calendar configuration on all FlexE Instances of the FlexE Group beginning with the same overhead frame in the overhead positions on each FlexE Instance.

When the FlexE demux is prepared to accept the switch of calendar configuration, it informs the FlexE mux by changing its CA bit to match the incoming CR bit on all FlexE Instances of the FlexE Group beginning with the same overhead frame in the overhead positions on each FlexE Instance. When this occurs is application specific. At the earliest, it occurs once the assignment of every calendar slot in the offline configuration has been received by the FlexE demux in a FlexE overhead frame with a good CRC after the change of the incoming CR bit. But the CA bit indication may be delayed for a variety of reasons: for example, software may need to be prepared for the incoming bandwidth change, for example in SDN applications.

The FlexE mux normally will switch calendar configurations only after receiving the CA bit acknowledgment after telling the FlexE demux it is ready to switch. The FlexE mux should set a timer (suggested default value as 1 second) after changing the outgoing CR bit. Appropriate values for the timer and action to be taken if the timer expires prior to receiving the CA bit are application specific. The timer could be as short as about 15ms (allowing for three complete transmissions of the calendar), but may be longer, for example, if software on the far end must be prepared to accept the switch to the updated calendar. The action to be taken on timer expiry without receiving the CA bit response is either to proceed with the switch or to raise an alarm and wait for corrective action to be taken. The FlexE mux indicates to the FlexE demux the change of calendar by changing the value of all three “C” bits to indicate the new calendar in the same FlexE overhead frame on all FlexE Instances.

Note that the availability of the above information in the protocol is not intended to limit the ways in which FlexE can be used. The FlexE demux may not act as a “slave” of the FlexE mux in terms of calendar configurations in every application. For example:

- A static configuration (e.g., one composed of a fixed number of FlexE Instances and PHYs, perhaps performing simple bonding for a single client, or supporting only a fixed calendar configuration for something like a port expander) would not need to fully implement this protocol. Such a configuration would simply transmit the A and B calendar configurations as fixed, always indicate the A calendar configuration as the calendar configuration in use, and would alarm the configuration inconsistency if the received calendar configuration from the far end is not the values expected or if the far end attempts to switch calendars (e.g., sends the calendar in use bits or the CR bit indicating calendar B rather than calendar configuration A).
- An application where a management system or SDN controller has access to the FlexE mux/demux at each end of the FlexE Group, that controller may configure the FlexE mux and demux configurations directly and instruct the two ends when to switch calendars. The information sent inband over the FlexE Instances within the PHYs might just be used as a check for the consistency of the configuration rather than as control for how the FlexE demux is configured.

7.3.5 Management Channel(s) and Synchronization Messaging Channel

Certain applications may require use of management channel(s). Two optional management channels and an optional Synchronization Messaging Channel are provided:

- A “section” management channel is carried from a FlexE Shim to the adjacent FlexE aware node (which may be the far end FlexE Shim in a simple router to router connection, or a FlexE aware transport network interface in the case of transport network does not terminate the FlexE Group).
- A “shim to shim” management channel which is carried end-to-end between the shims that terminate the FlexE Group.
- A synchronization messaging channel, which is applicable only for simple point-to-point bonding scenarios, e.g., creating a 300GbE equivalent by bonding three 100GBASE-R PHYs.

Each PHY in the FlexE Group can carry its own section and shim-to-shim management channels in the 50G FlexE Instance or the first 100G FlexE Instance on that PHY. The management channels are not aggregated across the FlexE Group. The synchronization messaging channel is only carried in the first FlexE Instance of the first PHY in the FlexE Group. For 200GBASE-R, 400GBASE-R, and 800GBASE-R PHYs,

block positions 4-8 in the FlexE overhead of the remaining 100G FlexE Instances are reserved, encoded with a data sync header (01), and zeros in the rest of the block as for any other reserved bits.

The format of the management channel is application specific. The only constraint on the management channel is that every 66B block is a legal format according to [802.3] clause 82. The protocol used over a management channel may be Ethernet based, using a combination of data and control blocks, or may be any other application-specific format using only data blocks. When a management channel is not used, it is transmitted as an Ethernet idle control blocks as illustrated in Figure 32.

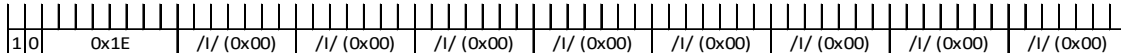


Figure 32: Ethernet Idle Control Block

The section management channel occupies two 66B blocks (blocks 4 and 5) of each FlexE overhead frame. The total capacity of the section management channel is approximately:

- 0.611 Mb/s (not counting the sync headers), or 0.630 Mb/s (counting the sync headers) per 50GBASE-R carrying a 50G FlexE Instance.
- 1.222 Mb/s (not counting the sync headers), or 1.260 Mb/s (counting the sync headers) per 100GBASE-R, 200GBASE-R, 400GBASE-R or 800GBASE-R, each carrying one or multiple 100G FlexE Instances.

The shim-to-shim management channel may occupy two or three 66B blocks of each FlexE overhead frame, depending on configuration.

For groups not configured to provide a synchronization messaging channel, the shim-to-shim management channel occupies blocks 6-8 of the FlexE overhead frame. The total capacity of the shim-to-shim management channel for this configuration is approximately:

- 0.916 Mb/s (not counting the sync headers), or 0.945 Mb/s (counting the sync headers) per 50GBASE-R carrying a 50G FlexE Instance.
- 1.833 Mb/s (not counting the sync headers), or 1.890 Mb/s (counting the sync headers) per 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R, each carrying one or multiple 100G FlexE Instances.

For groups configured to provide a synchronization messaging channel, the shim-to-shim management channel occupies blocks 7-8 of the FlexE overhead frame. The total capacity of the shim-to-shim management channel in this configuration is approximately:

- 0.611 Mb/s (not counting the sync headers), or 0.630 Mb/s (counting the sync headers) per 50GBASE-R carrying a 50G FlexE Instance
- 1.222 Mb/s (not counting the sync headers), or 1.260 Mb/s (counting the sync headers) per 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R, each carrying one or multiple 100G FlexE Instances.

When a group is provisioned to support a synchronization messaging channel, block position 6 of the FlexE overhead frame (on the first 50GBASE-R PHY, the first 100GBASE-R PHY, or the first 100G FlexE Instance on the first 200GBASE-R, 400GBASE-R, or 800GBASE-R PHY) is used to carry PTP and/or SSM messages as specified in [G.8264] or [1588]. These messages are 66B block encoded as per [802.3] clause 82. The interface timestamp point for a PTP event message transported over the synchronization

messaging channel shall be the start of the FlexE overhead multiframe preceding the first 66B block (the /S/ block) of the PTP event message.

Whether a group is configured to support a synchronization messaging channel is indicated in the Synchronization Configuration (SC) bit in the 1st block of the FlexE overhead frame in the first FlexE Instance. If the SC bit has the value 0, the shim-to-shim management channel occupies blocks 6-8 of the FlexE overhead frame on a in the first FlexE Instance. If the SC bit has the value 1, the shim-to-shim management channel occupies only blocks 7-8 of the FlexE overhead frame in the first FlexE Instance, and block 6 is allocated to the synchronization messaging channel. This is expected to be a static configuration aspect, and a synchronization messaging channel cannot be added or removed while the group is in service. A misconfiguration alarm can be raised if the SC bit persistently (e.g., 3 consecutive occurrences in FlexE overhead frames with good CRC) does not match the expected value for the configuration. The SC bit position in all but the first 100G FlexE Instance on a 200GBASE-R, 400GBASE-R, or 800GBASE-R PHY is reserved.

A potential application for the management channels is connectivity verification. When the LLDP protocol extension described in [FLEXE-ND] is used, the section management channel is used for that purpose. Other connectivity verification protocols (e.g. LLDP, per [802.1AB], without the [FLEXE-ND] extension) can use the section or shim-to-shim management channel.

When LLDP is used over the shim-to-shim or section management channels, the following apply:

- As there is no bridge in the middle of either the shim-to-shim or section management channels, all LLDPDUs shall use the “nearest bridge” destination MAC address 01-80-C2-00-00-0E.
- This IA does not define any new TLVs to describe the FlexE Group structure (e.g., PHY numbers and PHY map) as this information is fully contained in the FlexE overhead information.

7.3.6 FlexE Group Number

A 20-bit FlexE Group number is available to allow checking that the correct FlexE Instance is part of the correct FlexE Group.

The FlexE Group number is normally provisioned to the same value in both directions. All equipped 100G FlexE Instances on all PHYs of the FlexE Group must have the same FlexE Group number. It is not possible to have different 100G FlexE Instances on the same PHY be members of different FlexE Groups. Similarly, all 50G FlexE Instances in a FlexE Group of 50GBASE-R PHYs must have the same FlexE Group number.

The FlexE Group number is selected from the range 1-0xFFFFD. The values of 0x00000, 0xFFFFE, and 0xFFFFF are reserved and are not used to designate a FlexE Group. The received group number is checked against the provisioned group number and any mismatch will be alarmed to indicate the misconnection.

NOTE – the value 0xFFFFE is used by the neighbor discovery mechanism described in [FLEXE-ND].

7.3.7 Reserved Bits

The reserved bits in the FlexE overhead frame are reserved for possible future extensions to this implementation agreement. The reserved bits shall be transmitted as zero before scrambling. An implementation of this version of the IA should ignore these bits on receipt and leave the responsibility to an implementation of a newer version of the implementation agreement to recognize receipt of zeros

as an indication of interconnection with an older version, and presumably the newer version knows whether it is interoperable with the older version.

Note that one of the FlexE overhead bits was a reserved bit in FlexE 1.0 and has a specific use in FlexE 2.0 and beyond. The value “0” for this bit is compatible with the FlexE 1.0 mode of operation. Also the overhead byte carrying FlexE Payload Type indication in FlexE 3.0 and beyond, was reserved in FlexE 2.2 and prior versions.

7.3.8 Remote PHY Fault (RPF)

This is used to inform the far-end shim of a locally detected failure of the PHY. Since there is no 50G, 100G, 200G, 400G, or 800G RS layer per PHY, the FlexE overhead is used to convey this information. For 200GBASE-R, 400GBASE-R, and 800GBASE-R PHYs, the RPF bit is carried only in the FlexE overhead for the first 100G FlexE Instance on each PHY, and the RPF bit position is a reserved bit in the other 100G FlexE Instances. See clause 7.5.2.

Since RPF is an indication to the far-end shim, this indication is passed through and not processed at intermediate nodes in FlexE-Aware configurations as described in 8.3.

7.3.9 CRC-16

Primarily to avoid corrupting the content of the calendar configurations in the presence of bit errors, the FlexE overhead is protected by a CRC. The CRC is calculated over the following bits across the first three blocks of the FlexE overhead frame (in the order transmitted and received, not the order described):

- The D1, D2, and D3 bytes of the ordered set in overhead block 1.
- All eight octets after the sync header of overhead block 2.
- The first six octets after the sync header of overhead block 3.

The CRC is calculated using the polynomial $x^{16} + x^{12} + x^5 + 1$ with an initialization value of zero, where x^{16} corresponds to the MSB and x^0 corresponds to the LSB. This value is inserted by the FlexE mux into the transmitted overhead with the bit corresponding to x^{15} in the position transmitted first and the bit corresponding to x^0 transmitted last. Note that while this is the opposite of normal Ethernet bit-transmission order, it is consistent with the order of transmission of the Ethernet FCS. It is calculated by the FlexE demux over the same set of bits and compared to the received value. Various overhead described in the previous clauses is either accepted or ignored based on whether the CRC matches the expected value.

7.3.10 Payload Type

A one-byte FlexE payload type is defined to indicate the composition of the FlexE Group payload. The PT Field is located in bits 56 to 63 of the second overhead block of the overhead multiframe and its value is replicated in all equipped FlexE instances in the FlexE Group. The codepoints are defined in Table 2.

A new payload type is accepted if a new value of the PT field is received in an overhead frame with good CRC. The accepted payload type is checked against the provisioned payload type and any mismatch will be alarmed to indicate the misconfiguration. A mismatch is detected if the accepted and expected payload type does not match in one or more of the equipped 100G FlexE Instances in the group. The mismatch is cleared when the accepted and expected payload type matches in all of the equipped 100G FlexE Instances in the group. The mismatch condition is alarmed.

Table 2: FlexE payload types

Value	Payload
0x00	Reserved for FlexE versions earlier than 3.0 (could be 5G or 25G calendar slots)
0x01	FlexE 3.0 (or later) with 5G calendar slots
0x02	FlexE 3.0 (or later) with 25G calendar slots
0x03	FlexE 3.0 (or later) with 100G calendar slots
...	
0x80-0x8F	Reserved for proprietary use
...	
0xFE	PRBS test pattern
0xFF	Reserved

Implementations developed before version 3.0 of this IA do not support the payload type, and the byte is considered to be reserved (see 7.3.7).

7.4 FlexE Mux Data Flow

The FlexE Mux creates a logically serial stream of 66B blocks by interleaving FlexE Clients according to a calendar of length:

- 10n slots for a FlexE Group composed of n 50G FlexE Instances; or
- 20n slots for a FlexE Group composed of n 100G FlexE Instances.

Each slot corresponds to 5G of bandwidth. A FlexE Client is assigned a number of slots according to its bandwidth divided by 5G. The calendar configuration is distributed as described earlier in Figure 21.

Figure 33 presents an example of insertion of different bandwidth FlexE Clients into a logical calendar of length 10n and distribution to length 10 sub-calendars (50G FlexE Instances). The slots assigned to a particular FlexE Client do not all need to be on the same 50G FlexE Instance of the FlexE Group, and new clients can be added as long as there are sufficient slots available.

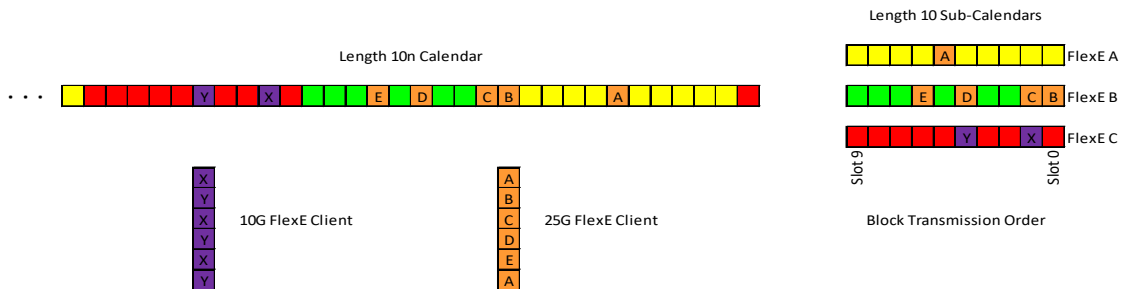


Figure 33: Illustration of Data Flow for FlexE mux based on 50G FlexE Instances with length 10 sub-calendars

Figure 34 presents an example of insertion of different bandwidth FlexE Clients into a logical calendar based on 100G FlexE Instances with 20 calendar slots per sub-calendar. The slots assigned to a particular FlexE Client do not all need to be on the same 100G FlexE Instance of the FlexE Group, and new clients can be added as long as there are sufficient slots available.

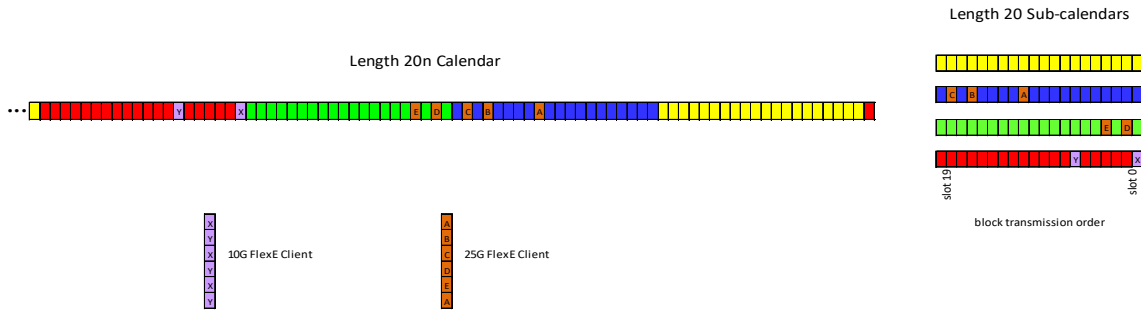


Figure 34: Illustration of Data Flow for FlexE Mux (5G calendar slots, 100G FlexE Instances)

Any slot in the calendar configuration which is either “unused” or “unavailable” will be filled with Ethernet Error control blocks with the format given in Figure 35. This ensures that any error in calendar slot assignment cannot appear to the FlexE demux as valid FlexE Client data.

1	0	0x1E	/E/ (0x1E)	/E/ (0x1E)	/E/ (0x1E)	/E/ (0x1E)	/E/ (0x1E)	/E/ (0x1E)	/E/ (0x1E)
---	---	------	------------	------------	------------	------------	------------	------------	------------

Figure 35: Ethernet Error Control Block Format

These rules allow for creation of the complete data sequence on each FlexE Instance of the FlexE Group, which are then padded and interleaved if necessary (see 6.2) to form the data sequence that is carried over each PHY. The FlexE overhead as described in clause 7.3 is inserted:

- onto each 50G FlexE Instance after every 2×1023 repetitions of the length 10 sub-calendar sequence in the same relative position to the calendar sequence on every 50G FlexE Instance of a FlexE group composed of m 50GBASE-R PHYs;
- onto each 100G FlexE Instance after every 1023 repetitions of the sub-calendar sequence in the same relative position to the calendar sequence on every 100G FlexE Instance of a FlexE group composed of m 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R PHYs.

This provides a marker which allows the data from the different 100G FlexE Instances of the FlexE Group to be re-interleaved in the original sequence so that the FlexE Clients can be extracted. The 66B block stream is then converted into the format for the individual FlexE Group PHY, which includes block distribution and alignment marker insertion, along with (if applicable) 256B/257B transcoding and FEC calculation and insertion.

In an implementation that restricts bandwidth allocation to 25G calendar granularity, the FlexE mux may interleave clients according to a calendar of 2n slots for a group composed of n 50G FlexE Instances, or 4n slots for a group composed of n 100G FlexE Instances. Each slot corresponds to 25G of bandwidth. A FlexE Client is assigned a number of slots according to its bandwidth divided by 25G. A 25G calendar slot occupies five consecutive 66B blocks within a FlexE Instance. Note that these slots do not remain

adjacent when multiple 100G FlexE Instances are 66B block interleaved on a 200GBASE-R, 400GBASE-R, or 4x66B block interleaved on a 800GBASE-R PHY. An illustration of 25G calendar slot allocation showing a logical length 4 sub-calendar per 100G FlexE Instance is provided in Figure 36. Appendix B illustrates how the 66B blocks from 25G calendar slots distribute when 100G FlexE Instances are interleaved for 200G or 400G PHYs.

As with 5G calendar slots, 25G calendar slots that are unused or unavailable are filled with (in groups of five) error control blocks as indicated in Figure 35. This ensures that any errors in calendar slot assignment cannot be interpreted by the FlexE demux as valid client data.

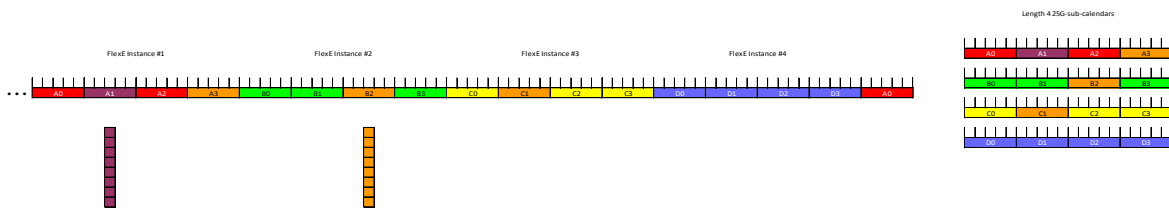


Figure 36: Illustration of Data Flow for FlexE Mux (four 25G calendar slots per 100G FlexE Instance)

In an implementation that restricts bandwidth allocation to 100G calendar granularity, the FlexE mux may interleave clients according to a calendar of n slots for a group composed of n 100G FlexE Instances. Each slot corresponds to 100G of bandwidth. A FlexE Client is assigned a number of slots according to its bandwidth divided by 100G. A 100G calendar slot occupies the twenty consecutive 66B blocks of a 100G FlexE Instance. Note that these slots do not remain adjacent on the PHY bit stream when multiple 100G FlexE Instances are 66B block interleaved on a 200GBASE-R or 400GBASE-R PHY or 4x66B block interleaved on an 800GBASE-R PHY. An illustration of 100G calendar slot allocation showing a logical length 1 sub-calendar per 100G FlexE Instance is provided in Figure 37 .

As with 5G calendar slots, 100G calendar slots that are unused or unavailable are filled with (in groups of twenty) error control blocks as indicated in Figure 35. This ensures that any errors in calendar slot assignment cannot be interpreted by the FlexE demux as valid client data.

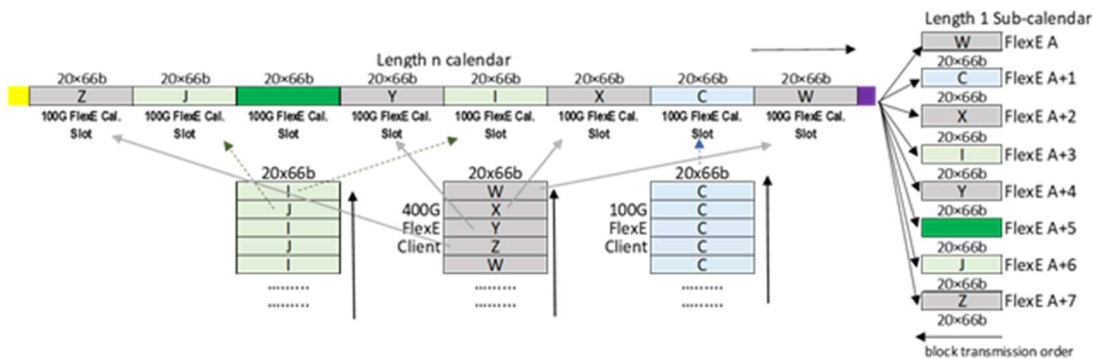


Figure 37 : Illustration of Data Flow for FlexE Mux (single 100G calendar slot per 100G FlexE Instance)

7.5 FlexE Demux Data Flow

The FlexE Demux operates on a sequence of 66B blocks received from each FlexE Instance of the FlexE Group. Recovering this sequence of blocks includes (if applicable), FEC error correction and FEC remove and trans-decoding to 64B/66B, PCS or FEC lane alignment, reinterleaving, and alignment marker removal. If necessary, pad blocks are removed and multiple 100G FlexE Instances present on the PHY are dis-interleaved (see 6.2).

Figure 38 illustrates the interleaving of calendar slots from 50G FlexE Instances in master-calendar order for the extraction of FlexE clients from the group.

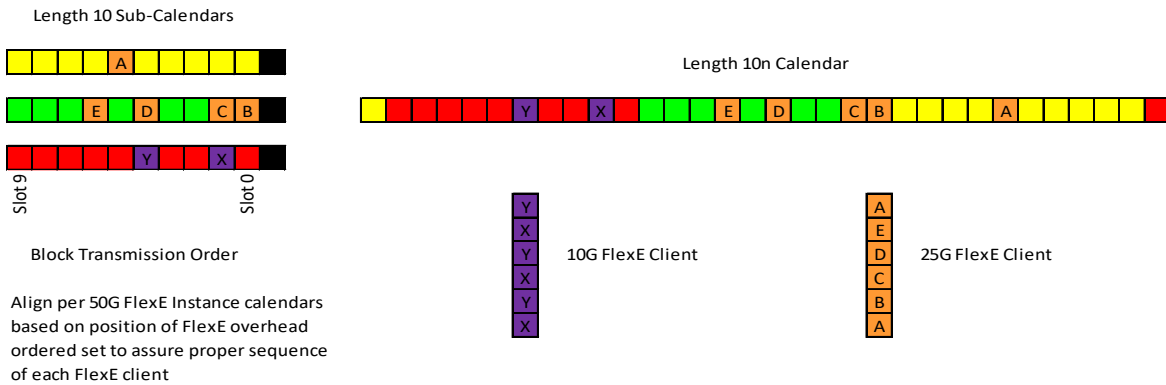


Figure 38: Illustration of FlexE Demux Data Flow (5G calendar slots from length 10 sub-calendars on 50G FlexE Instances)

Figure 39 illustrates the interleaving of calendar slots from 100G FlexE Instances recovered from 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R PHYs in master-calendar order for the extraction of FlexE clients from the group.

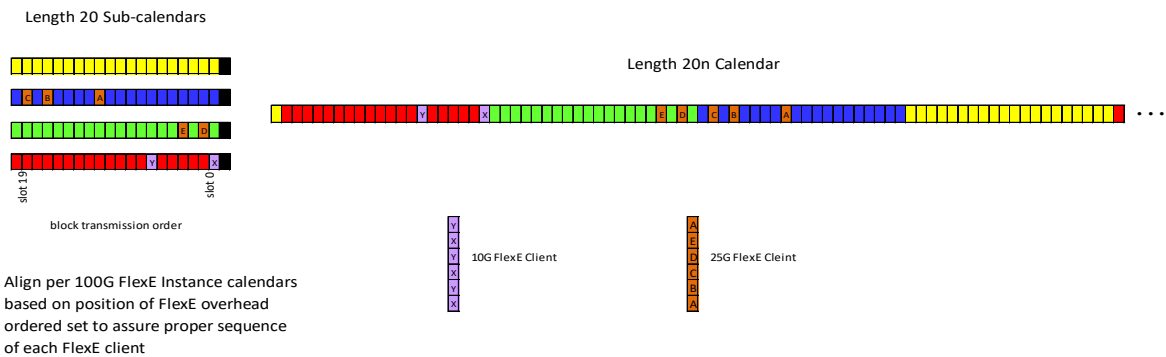


Figure 39: Illustration of FlexE Demux Data Flow (5G calendar slots, 100G FlexE Instances)

Note that the FlexE overhead frame repeats:

- For 50G FlexE Instances, on a cycle of approximately 209.52μs, which allows for measuring the skew differences between 50G FlexE Instances recovered from the 50GBASE-R PHYs of the FlexE Group of approximately ±104μs.

- For 100G FlexE Instances, on a cycle of approximately 104.76μs, which allows measuring skew differences between 100G FlexE Instances recovered from the 100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R PHYs of the FlexE Group of approximately ±52μs.

In an implementation that restricts bandwidth allocation to 25G calendar granularity, the FlexE demux may extract clients from the recovered 66B block stream according to a calendar of 2n slots for a group composed of 50G FlexE Instances, or 4n slots for a group composed of n 100G FlexE Instances. Each slot corresponds to 25G of bandwidth. A 25G calendar slot occupies five consecutive 66B blocks within a 50G or 100G FlexE Instance. Note that these slots are not adjacent on a PHY when multiple 100G FlexE Instances are 66B block interleaved on a 200GBASE-R, 400GBASE-R, or 800GBASE-R PHY. An illustration of FlexE Clients extracted based on 25G calendar slot allocation is provided in Figure 40.

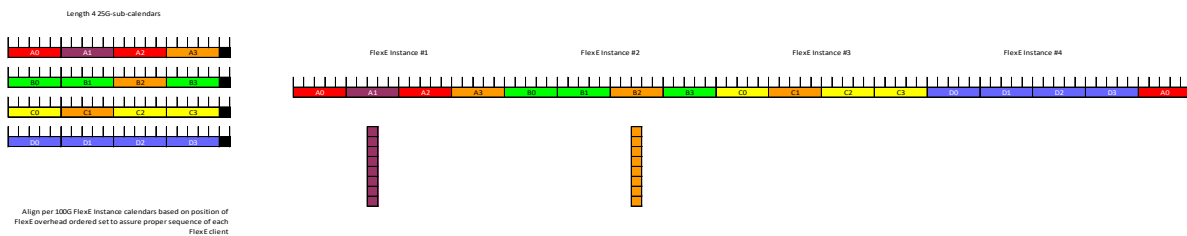


Figure 40: Illustration of Data Flow for FlexE Demux (25G calendar slots from length 4 sub-calendars on 100G FlexE Instances)

In an implementation that restricts bandwidth allocation to 100G calendar granularity, the FlexE demux may extract clients from the recovered 66B block stream according to a calendar of n slots for a group composed of n 100G FlexE Instances. Each slot corresponds to 100G of bandwidth. A 100G calendar slot occupies twenty consecutive 66B blocks within a 100G FlexE Instance. Note that these slots are not adjacent on a PHY when multiple 100G FlexE Instances are 66B block interleaved (before transcoding) on a 200GBASE-R or 400GBASE-R PHY or are 4x66B block interleaved (before transcoding) on a 800GBASE-R PHY. An illustration of FlexE Clients extraction based on 100G calendar slot allocation is provided in Figure 41.

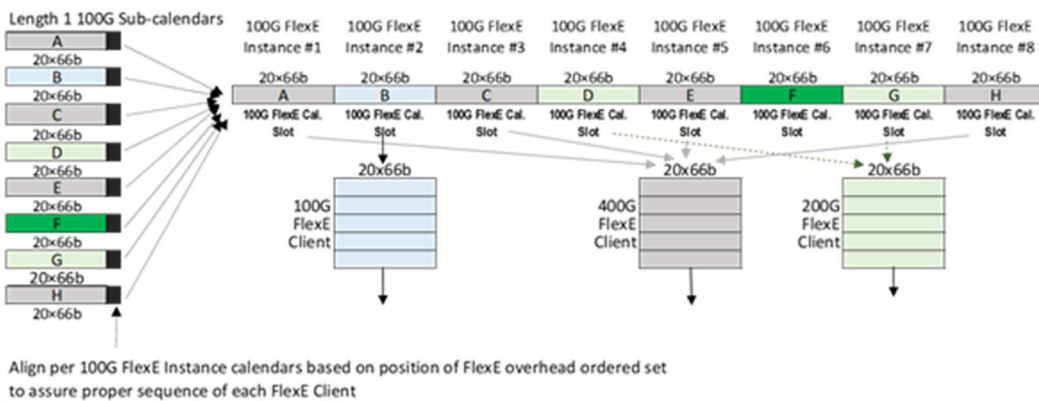


Figure 41 : Illustration of Data Flow for FlexE Demux (100G calendar slots from length 1 sub-calendars on 100G FlexE Instances)

7.5.1 Skew Tolerance Requirements

The amount of skew to be expected between the PHYs of the FlexE Group are application specific. Note that skew does not accumulate between 100G FlexE Instances interleaved onto the same PHY, so deskew will be performed across 100G FlexE Instances from different PHYs that have skew between them. This implementation agreement specifies skew requirements for two classes of applications.

Low Skew Applications include intra-data-center applications, plus those transport network applications where the FlexE Shim is implemented in the transport equipment and the FlexE Clients rather than the PHYs of the FlexE Group are carried across the transport network. The skew tolerance requirement for low skew applications is 300ns. Note that the intra-PCS-lane skew tolerance requirement for 50GBASE-R, 100GBASE-R, 200GBASE-R, 400GBASE-R is 180ns, and 800GBASE-R is 152ns. A larger skew budget is established for FlexE applications of similar reach to account for the fact that the PCS lane deskew is not synchronized across the PHYs of the FlexE Group, and there may be other variation, such as cable length, or even heterogeneous Ethernet PHY types which are not present within a single Ethernet interface.

High Skew Applications include the broadest range of transport network applications where the PHYs of the FlexE Group rather than the FlexE Clients are carried over the transport network (FlexE aware/unaware transport). The skew tolerance for high skew applications may need to be as high as 10 μ s. This is established to account for about 6 μ s of dispersion-related skew if the PHYs are mapped over lambdas at opposite ends of the “C” band over large distances (e.g., trans-Pacific), with extra margin for things like split-band amplifiers and patch cords or the processing time to crunch and uncrunch the signal in the case where not all of the calendar slots can be carried over the transport network connection.

7.5.2 FlexE Demux Fault Handling

If the inter-PHY skew (detected when attempting to deskew 50G or 100G FlexE Instances on different PHYs) exceeds the skew tolerance of the implementation, the FlexE Clients will not be demapped from the incoming PHYs, but will be sent continuous Ethernet Local Fault Ordered sets as illustrated in Figure 42 at the FlexE Client rate.

If one or more of the PHYs of the FlexE Group has failed (e.g., loss of signal, failure to achieve block lock or alignment lock, hi BER, any other condition that results in PCS_status=FALSE, or loss of FlexE overhead frame lock or multi-frame lock), the Remote PHY fault bit is set to one in the reverse direction on that PHY in the 50G FlexE Instance or the first 100G FlexE Instance carried on that PHY. Note that this must be done in the FlexE overhead since there is no normal PHY-rate RS layer available to indicate remote fault to the far end. In addition, even if none of the PHYs of the FlexE Group have failed, if one or more of the FlexE Instances fails to achieve overhead frame lock or overhead multiframe lock, if there is inconsistency among the FlexE maps, FlexE Instance numbers, or FlexE Group numbers, or, for FlexE 3.0 and beyond implementations, among FlexE Payload Type indications, received on the different FlexE Instances of the group, or if the skew between FlexE instances from different PHYs exceeds the deskew buffer provided by the implementation, an appropriate alarm should be raised and all of the FlexE Clients will be sent continuous Ethernet Local Fault Ordered sets as illustrated in Figure 42 at the FlexE Client rate.

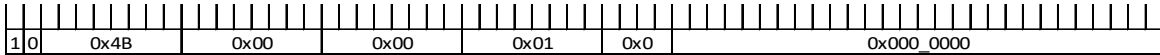


Figure 42: Ethernet Local Fault Ordered Set

7.6 FlexE Group Configuration

When a new FlexE Group is brought into service, the initial configuration must be provisioned from both ends, and the initial configuration must be the same. The group is configured to consist of either:

- from 1 to n 50G FlexE Instances, each carried over a 50GBASE-R PHY; or
- from 1 to n 100G FlexE Instances carried over from 1 to m PHYs of the same rate (100GBASE-R, 200GBASE-R, 400GBASE-R, or 800GBASE-R).

All of the PHYs which are configured as part of the group are brought into service, all transmitting the stream of 66B blocks including the FlexE overhead of the FlexE Instances carried over that PHY. See 6.2 concerning the manner in which FlexE Instances are carried over PHYs. The respective bits of the “FlexE Map” field are set to indicate which FlexE Instances (and by extension, which PHYs) the near end has configured as part of the group. The group is brought into service when all incoming PHYs are up, receiving FlexE overhead on each of the expected FlexE Instances within the allowable skew tolerance of each other, and the received FlexE Instances all indicate the expected FlexE Instances as being part of the group. When the FlexE Group is first configured and brought into service, all of the calendar slots are normally set to “unused” or “unavailable” (as needed for partial-rate transport configurations), although there may be certain implementations such as a static configuration supporting simple bonding which are brought up with a single FlexE Client filling all available calendar slots.

7.7 Energy Efficient Ethernet (EEE)

EEE is not supported on the PHYs of a FlexE Group, as there is no good way to verify that every FlexE Client is idle and put the entire group into a low power idle mode.

The “Fast Wake” mode of EEE can be supported for a FlexE Client. The LPI control characters used during Fast Wake can simply pass through the 66B payload block positions allocated to that client by the calendar configuration, allowing an implementation to be aware when data is not arriving on a given FlexE Client.

For FlexE Clients generated internally to the system or created from an optical Ethernet PHY, “Fast Wake” is the only mode of operation possible, and negotiation between the FlexE Client endpoints (above the FlexE Client MAC) to confirm the use of “Fast Wake” will occur via LLDP. For a FlexE Client generated from a copper Ethernet PHY, the AN capabilities should always indicate “No” for both “Deep Sleep” and “Fast Wake”, and the “Fast Wake” mode can then be enabled via LLDP negotiation.

7.8 FlexE test patterns

A framed FlexE PRBS test pattern is used for validating FlexE protocol framing, overhead, and adaptation to the lower sublayers of the Ethernet PHY without user traffic being present. The pattern is PRBS31 per [IEEE 802.3] with the initial generator state (see Figure 43) of all 1s.

The test pattern is carried in every equipped FlexE Instance. The pattern is inserted by the FlexE mux in the 64 payload bits of the data blocks between FlexE overhead blocks as shown in Figure 24 and Figure 25. The FlexE demux extracts the bits and verifies the PRBS31 sequence.

The presence of the framed PRBS test pattern is identified by the payload type as shown in Table 2.

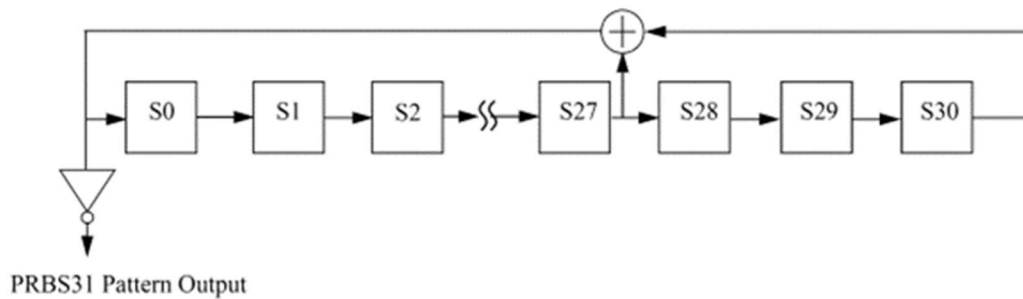


Figure 43: Framed PRBS test pattern generator

Note: Implementations developed before version 3.0 of this IA may not support the test pattern feature.

8 Transport Network Mappings for Flex Ethernet Signals

Three different methods of mapping of FlexE signals over transport networks are possible.

8.1 FlexE Unaware Transport

The case of FlexE unaware transport involves the transport network mapping each of the Ethernet PHYs independently over the transport network using the existing PCS codeword transparent mapping. Since the FlexE mux and FlexE demux are separated by transport network distances, this requires a “high skew” implementation of the FlexE Shim as described in clause 7.5.1.

Note that certain existing OTN mappers/demappers are not fully PCS codeword transparent with respect to LF and RF ordered sets, and may mistake an LF or RF sent for an individual FlexE Client as an LF or RF for the entire link and bring the link down. This is not an issue in the case of FlexE unaware transport of simple bonding to carry a larger rate flow, as failure of the single high-rate FlexE Client is equivalent to a failure of the group. But it may be an issue if FlexE unaware transport is used to carry a group of multiple lower-rate FlexE Client services using less than fully PCS codeword transparent mappings.

8.2 FlexE termination in the Transport

The next case is where the FlexE Shim is terminated by the transport network equipment, and rather than carrying the PHYs of the FlexE Group over the transport network, the FlexE Clients are carried over the transport network. The rate-adapted FlexE Client bit rate is given in clause 6.4.

Note that since the stream of blocks presented to the transport network from the FlexE Shim does not have any timing information, the transport network is not required to transport the signal at the exact adapted FlexE Client bit-rate: idle insertion/deletion or padding may be used in the mapping specified by ITU-T if it provides a more convenient rate for the transport.

When a FlexE Client is mapped in this manner, it may be connected, at the OTN egress, to another FlexE Shim where it will be clock aligned with the FlexE Group at the network egress. It may also be connected to an Ethernet PHY with the same nominal MAC rate as the FlexE Client using the appropriate conversion as described in clause 7.2.2.

Note that in the case where it is necessary to connect a FlexE Client to an Ethernet PHY across an OTN where the Ethernet PHY uses a legacy mapper/demapper, it may be necessary to perform the conversion of the FlexE Client to the Ethernet PHY format according to clause 7.2.2 immediately after the FlexE Shim and to map the FlexE Client over OTN as if it were an Ethernet PHY of the corresponding rate.

This second case can use a “low skew” implementation of the FlexE Shim as described in clause 7.5.1.

8.3 FlexE Aware Transport

The third case is where the transport network equipment is aware that the information of a FlexE Group is being carried, but does not terminate the FlexE Group in the transport network equipment.

This is typically used to support cases where the transport network capacity is less than the aggregate capacity of all Ethernet PHYs of the FlexE Group, the FlexE PHY rate is greater than the transport network container rate, the transport network container rate is not an integral multiple of the PHY rate, or there is a reason (for example, wavelengths terminated on different transponder line cards) that it is not possible to terminate the FlexE Shim in the transport equipment.

For such cases, the following aspects of the FlexE Group are negotiated between the transport network operator and the customer requesting transport of the FlexE Group, based on the FlexE Client bandwidth that needs to be carried and the capacity of the transport network containers that the transport operator can provide:

- The number and bit rate of the FlexE PHYs
- In the case of 200G, 400G, or 800G PHY rate, the total number of equipped 100G FlexE Instances in the Group and the number of equipped instances on each PHY (for 50G or 100G PHYs, each PHY carries one Instance so there are no unequipped Instances)
- The number of available calendar slots for each equipped instance
- The number and bit rate of the transport network containers that will be used, and the assignment of FlexE PHYs to those containers.

Appendix D shows an example of a FlexE aware transport configuration.

The transport network equipment terminates the section management channel of each PHY. The information from this channel is extracted, and the section management channel is not used over the transport network. At egress, a new section management channel is inserted on each PHY.

The transport network equipment will discard Unequipped Instances and may “crunch” a FlexE Instance of the FlexE Group by allowing bits or bytes to be discarded from the unavailable calendar slots at the transport network ingress. Unequipped Instances and “crunched” bits or bytes are re-inserted with fixed values at the transport network egress. The mapping of this requires serializing and deskewing the PCS lanes of the PHY, recovering the FlexE Instance(s) from the PHY, discarding the unequipped instances, then discarding from the “UNAVAILABLE” calendar slots from the equipped instances to reduce the bit rate. For example, if only 15 of 20 calendar slots are available in a sub-calendar for a given 100G FlexE Instance, after discarding the 5 unavailable slots, there are effectively 1023 repetitions of a length 15 calendar that must be transported. At the transport network egress, the bits or bytes removed from the unavailable slots are restored to the FlexE Instance stream of 66B blocks so that error control blocks occur in every unavailable slot as illustrated in Figure 35.

For FlexE Aware Transport supported by the OTN, the FlexE Group and OTN configurations at both ends of the transport network are required to be the same, and each equipped 100G FlexE Instance on every PHY is required to have at least one available calendar slot. It is expected that the granularity of partial-rate transport is 25 Gb/s for 50G, 100G, 200G, and 400G PHYs, and 100 Gb/s for 800G PHYs. It is expected that 100 Gb/s granularity for partial rate transport will be implemented via the use of unequipped instances only (i.e. there will not be any unavailable calendar slots).

As described in clause 6.6, unavailable slots are always at the end of the sub-calendar configuration for the respective FlexE Instance. A partially filled 100G FlexE Instance should normally be the last equipped 100G FlexE Instance on a PHY. The rate of a wavelength is not expected to change in service. Therefore, when a partial-rate signal is carried over the OTN, the transport network equipment is statically configured to drop bits or bytes from any unavailable calendar slots, and to drop any unequipped 100G FlexE instances. The transport network equipment will restore the equivalent set of PHYs at the OTN network egress by inserting any unequipped 100G FlexE instances that were discarded at the OTN network ingress and inserting error control blocks as shown in Figure 34 into each unavailable calendar slot. The transport network equipment is not expected to dynamically react to which FlexE Instances are configured as unequipped or which calendar slots are marked as unavailable in the calendar configurations, but may non-intrusively monitor the FlexE overhead and detect an error condition if an instance that is expected to be unequipped is equipped (detected based on the FlexE Group ID, see 6.3) or a calendar slot the transport network equipment has been configured to drop indicates that it is carrying FlexE Client data rather than being marked as unavailable.

Note that this IA provides only an overview of FlexE aware transport. Details of the mapping into OTN are specified in [G.709].

9 Appendix A: Test Vectors

Test vectors for the first three overhead blocks of the FlexE overhead frame are given in Figure 44, Figure 45, Figure 46, and Figure 47. Note that the last five overhead blocks of the FlexE overhead frame form the two management channels and/or synchronization messaging channel. The positions of the 136 bits as coefficients of the polynomial across which the CRC is calculated is indicated. For example, the C bits in overhead blocks 1, 2, and 3 correspond to the coefficients of x^{135} , x^{111} , and x^{47} as shown in Figure 44, Figure 45, Figure 46, and Figure 47, respectively.

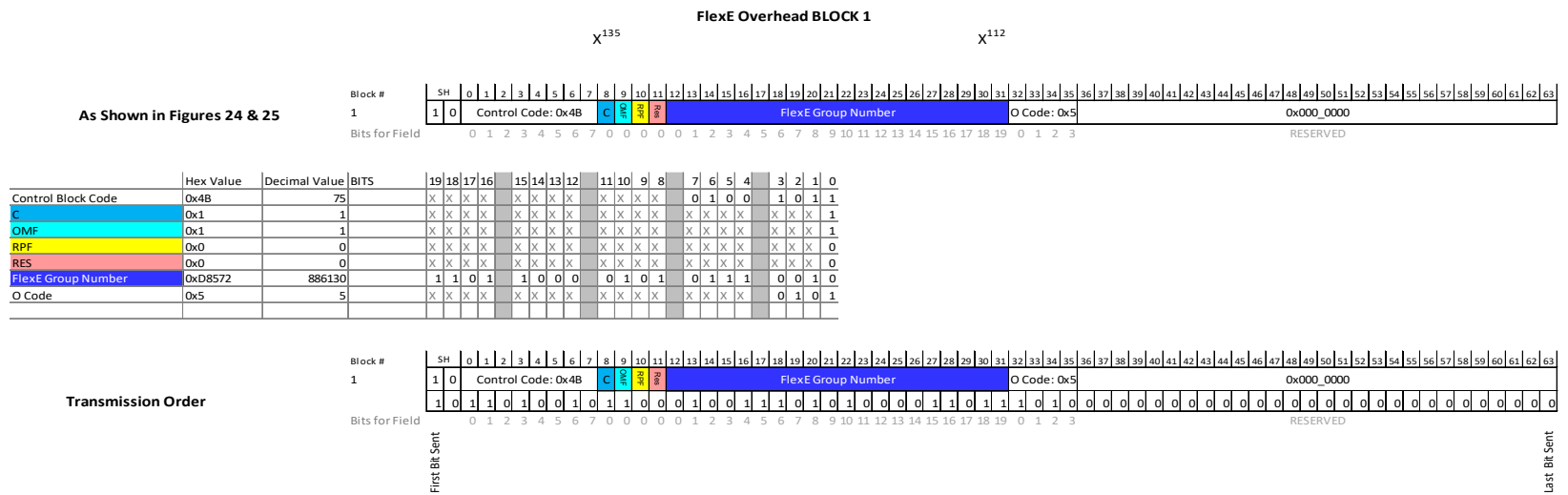


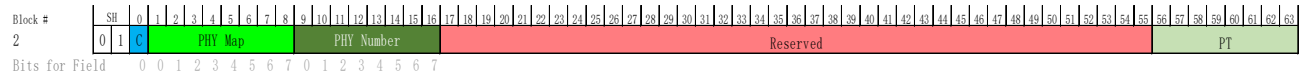
Figure 44: Test Vector for first block of FlexE overhead frame

FlexE Overhead BLOCK 2

X¹¹¹

X⁴⁸

As Shown in Figures 24 & 25



	Hex Value	Decimal Value	BITS	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C	0x1	1		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	
PHY Map	BITMAP	BITMAP		X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0	1	1	1	0
PHY Number	0x80	128		X	X	X	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	
RES	0x0000000	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
PT	0x03																							

{PHY7 = 0; PHY6 = 0; PHY5 = 0; PHY4 = 0; PHY3 = 1; PHY2 = 1; PHY1 = 1; PHY0 = 0;}

Not All bits shown

Transmission Order

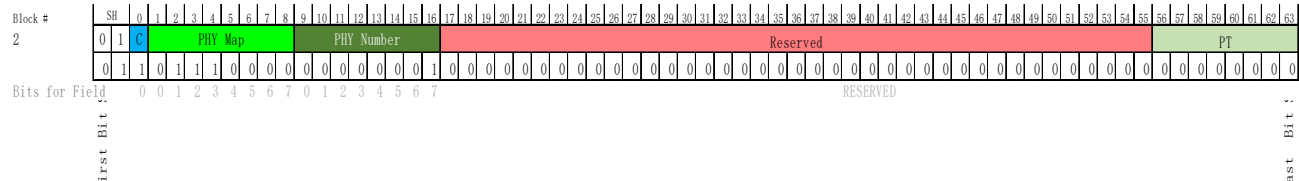


Figure 45: Test Vector for second block of FlexE overhead frame with PT codepoint 0x03

10 Appendix B: (Informative) Illustration of 25G Calendar Slot Distribution across 200G or 400G PHYs

Figure 48 and Figure 49 illustrate the distribution of 25G Calendar slots across 200G and 400G PHYs respectively at the FlexE mux.

Figure 50 and Figure 51 illustrate the corresponding behavior at the FlexE demux. It is important to note that while a 25G calendar slot is composed of 5 adjacent 66B block positions within a logical 100G FlexE Instance, these block positions do not remain adjacent when multiple 100G FlexE Instances are interleaved on a FlexE PHY.

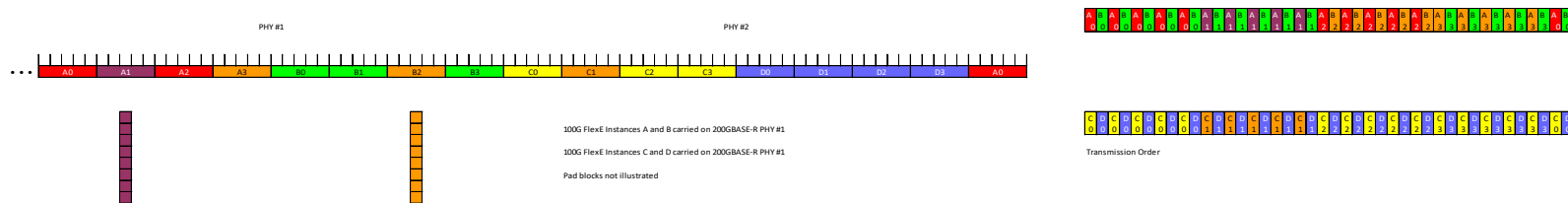


Figure 48: Example of 25G Calendar Slots Multiplexed onto 200G FlexE PHYs

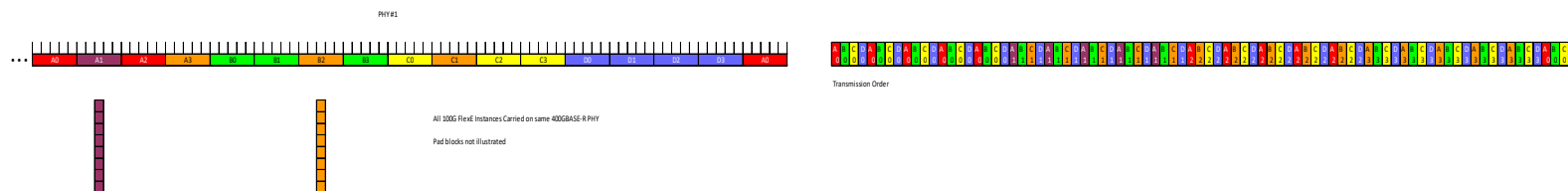


Figure 49: Example of 25G Calendar Slots Multiplexed onto a 400G FlexE PHY

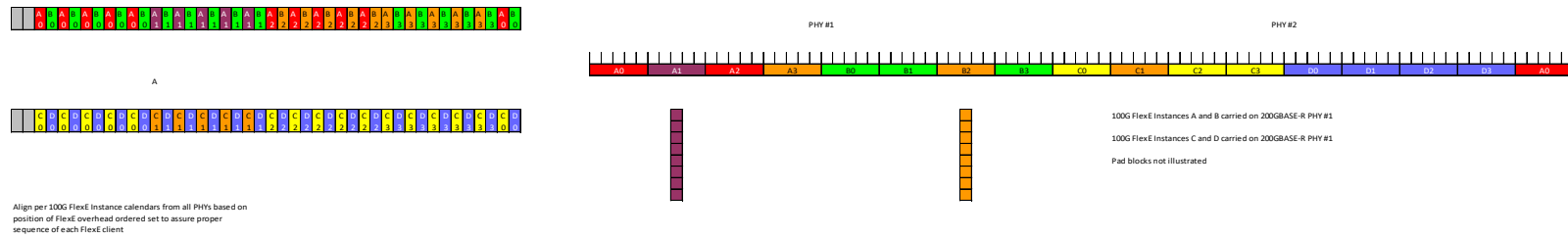


Figure 50 : Example of 25G Calendar Slots Demultiplexed from 200G FlexE PHYs

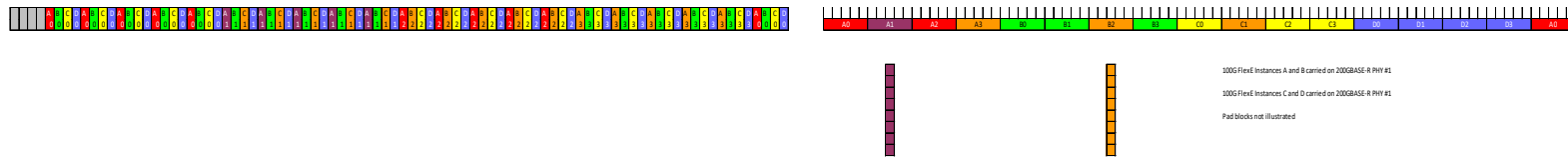


Figure 51 : Example of 25G Calendar Slots Demultiplexed from a 400G FlexE PHY

11 Appendix C: (Informative) FlexE Client Synchronization

Clause 7.3.5 describes the mechanism for transporting synchronization information via the FlexE Group. The entities being synchronized are the FlexE Shims at each end of the group.

An additional application that may be useful in some implementation is to provide synchronization between MAC clients, where the client streams are transported over a FlexE Group. The application scenario is illustrated in Figure 52.

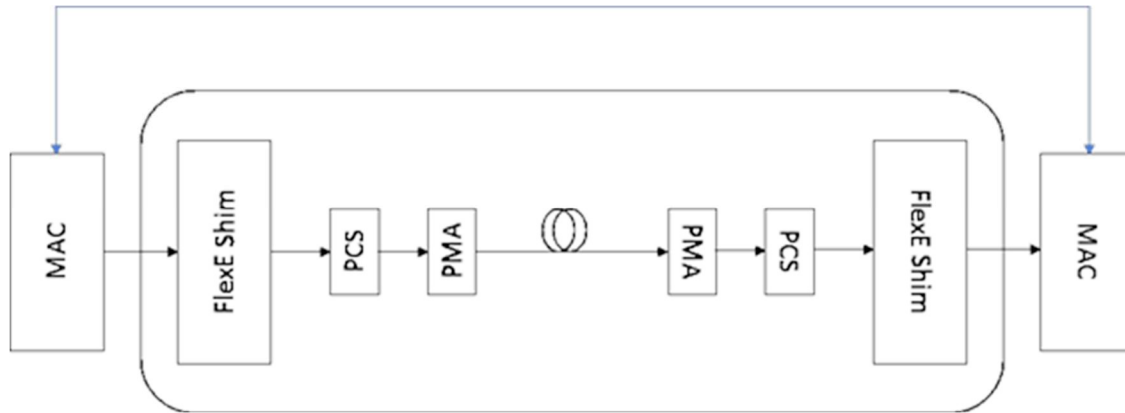


Figure 52: Illustration of exchange of PTP event messages between MAC clients

In [802.3] and [1588], the interface time stamp point for a PTP event message of a normal Ethernet client would be the time when the beginning of the first symbol after the SFD of the PTP event message passes the reference plane. When Ethernet streams are carried over a FlexE Group as FlexE Clients, additional considerations are required to emulate normal Ethernet PTP behavior. For example, a complication of an implementation that does this is that since the PTP message is sent in the FlexE Client stream, the actual position of the SFD at the MDI could be on any lane of any PHY of the FlexE Group. In the transmit direction, this implies that after the PTP messages are inserted into the client stream, the SFD position must be marked, and the marking of that position must be carried through the FlexE PHY distribution and the per PHY PCS lane distribution of each PHY so that the timestamps can be taken when the markers are detected at the PMA. In the receive direction, candidate timestamp positions (any SFD occurrence) must be recorded on each group of received data on each PHY, and this information must be propagated up through the FlexE demux so that once the FlexE Client stream is reassembled and the PTP message can be identified, the corresponding timestamp position can be identified. How to carry out this processing in an implementation is outside of the scope of this Implementation Agreement.

12 Appendix D: (Informative) FlexE Aware Configuration Example

As noted in 8.3, for FlexE aware transport, the maximum transport container capacity is determined by the transport network operator, while the required FlexE Client bandwidth to be carried is determined by the customer. The structure of the FlexE Group is negotiated between the customer and transport network operator in a manner that allows both of those constraints to be satisfied. The FlexE Group is divided into a number of subgroups that comprise an integer number of FlexE PHYs. Each subgroup is associated with one transport network container. The FlexE Client capacity of each subgroup is aligned to the transport container capacity by configuring unequipped instances and unavailable slots, as appropriate.

Table 3 shows an example of a FlexE Group with 700G of capacity that will be carried over 250G transport network containers. In this example, the FlexE Group is divided into three equal-size FlexE Sub-Groups, each containing $3 \times 100\text{G}$ FlexE Instances (with a total of 50 available calendar slots and 10 unavailable calendar slots). This FlexE Group could use 100G, 200G, or 400G PHYs.

Note that there are other configurations that are not shown in the example.

Table 3 – FlexE Aware Transport configuration example

FlexE Client bandwidth	700G		
Maximum transport container bit rate	250G		
Number of FlexE Sub-Groups required	$\lceil 700 / 250 \rceil = 3$		
Number of 100G FlexE Instances per FlexE Sub-Group	$\lceil 250 / 100 \rceil = 3$		
Number of 100G FlexE Instances in FlexE Group	$3 \times 3 = 9$		
Number of available calendar slots per FlexE Sub-Group	$\lceil 250 / 5 \rceil = 50$		
Number of unavailable calendar slots per FlexE Sub-Group	$3 \times 20 - 50 = 10$		
Available calendar slots per 100G FlexE Instance in FlexE Sub-Group	20, 20, 10		
FlexE PHY rate	100G	200G	400G
Number of FlexE PHYs per FlexE Sub-Group	3	$\lceil 3 / 2 \rceil = 2$	$\lceil 3 / 4 \rceil = 1$
Number of FlexE PHYs in FlexE Group	$3 \times 3 = 9$	$3 \times 2 = 6$	$3 \times 1 = 3$
Number of unequipped 100G FlexE Instances per FlexE Sub-Group	N/A	1	1

13 References

13.1 Normative references

- [802.3] IEEE Std 802.3™-2022 *Standard for Ethernet*.
- [802.3df] IEEE Std. 802.3df-2024, Amendment 9: Media Access Control Parameters for 800 Gb/s and Physical Layers and Management Parameters for 400 Gb/s and 800 Gb/s Operation
- [G.709] ITU-T Recommendation G.709 (06/2020), Amd 3 (03/2024), *Interfaces for the Optical Transport Network*.
- [G.8264] ITU-T Recommendation G.8264 (08/2017), Amd 1 (03/2024), *Distribution of timing information through packet networks*.
- [G.8023] ITU-T Recommendation G.8023 (06/2018), Amd 2 (03/2024), *Characteristics of equipment functional blocks supporting Ethernet physical layer and Flex Ethernet interfaces*.
- [802.1AB] IEEE Std 802.1™AB-2016, Station and Media Access Control Connectivity Discovery.
- [1588] IEEE Std 1588-2008, IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems
- [FLEXE-ND] OIF FlexE Neighbor Discovery Implementation Agreement (09/2018)

14 Appendix E: List of companies belonging to OIF when document is approved

Accelight Technologies, Inc.	H3C Technologies Co., Ltd.	Ranovus
Accton Technology Corporation	Hakusan Inc	Retym
Adtran Networks SE	Hewlett Packard Enterprise (HPE)	Rosenberger
Advanced Fiber Resources (AFR)	HGGenuine Optics Tech Company	Hochfrequenztechnik GmbH & Co. KG
Advanced Micro Devices, Inc.	Hirose Electric Co. Ltd.	Ruijie Networks Co., Ltd.
AIO Core Co., Ltd	Hisense Broadband Multimedia Technologies Co., LTD	Samsung Electronics Co. Ltd.
Alibaba	Huawei Technologies Co., Ltd.	Samtec Inc.
Alphawave Semi	InfiniLink	SCINTIL Photonics
Amazon	Integrated Device Technology	Semtech Canada Corporation
Amphenol Corp.	Intel	Senko Advanced Components
Anritsu	Juniper Networks	SeriaLink Systems Ltd.
Applied Optoelectronics, Inc.	Kandou Bus	Sicoya GmbH
Arista Networks	KDDI Research, Inc.	SiFotonics Technologies Inc.
Astera Labs	Keysight Technologies, Inc.	Silith Technology
ATOP Corporation	KYOCERA Corporation	Socionext Inc.
Ayar Labs	Lessengers Inc.	Source Photonics, Inc.
BitifEye Digital Test Solutions GmbH	Lightmatter	Spirent Communications

BizLink Technology, Inc.	Linktel Technologies Co., Ltd.	RAM Photonics Industrial, LLC
Broadcom Inc.	Lumentum	Sumitomo Electric Industries, Ltd.
Cadence Design Systems	Lumiphase AG	Sumitomo Osaka Cement
Casela Technologies USA	LUXIC Technology Co	Synopsys, Inc.
Celero Communications Inc.	Luxshare Technologies International, Inc.	TE Connectivity
Celestica	MACOM Technology Solutions	Tektronix
China Telecom	Marvell Semiconductor, Inc.	Telefonica S.A.
CICT	MaxLinear Inc.	TELUS Communications, Inc.
Ciena Corporation	MediaTek	TeraHop US
Cisco Systems	Meta Platforms	Teramount
Coherent	Microchip Technology Incorporated	TeraSignal, LLC.
ColorChip LTD	Microsoft Corporation	Texas Instruments
Cornelis Networks, Inc.	Mitsubishi Electric US, Inc.	US Conec
Corning	Molex	Viavi Solutions Deutschland GmbH
Credo Semiconductor (HK) LTD	Multilane Inc.	Wilder Technologies, LLC
CUBIQ Technologies	NEC Corporation	Wistron Corporation
Dai Nippon Printing Co., Ltd.	New Photonics, Ltd.	Xphor Ltd.
Dell, Inc.	Nokia	Yamaichi Electronics Ltd.
Dexerials Corporation	NTT Corporation	ZTE Corporation
DustPhotonics	Nubis Communications, Inc.	
EFFECT Photonics B.V.	NVIDIA	
Eoptolink Technology	O-Net Technologies (Shenzhen) Group Co., Limited	
Epson Electronics America, Inc.	Omatrix Ltd Co	
Ericsson	Omniva LLC	
EXFO	Optomind Inc.	
Fabrinet	Orange	
Foxconn Interconnect Technology Ltd	PETRA	
Fujikura	Point2 Technology	
Fujitsu	Precision Optical Technologies	
Furukawa Electric Co., Ltd.	Quantifi Photonics USA Inc.	
Global Foundries	Quintessent Inc.	
Google		