ITUPublications
Recommendations

International Telecommunication Union
Standardization Sector

Recommendation

ITU-T G.709.3/Y.1331.3 (2020) Amd. 1 (11/2022)

SERIES G: Transmission systems and media, digital systems and networks

Digital terminal equipments – General

SERIES Y: Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities

Internet protocol aspects – Transport

Flexible OTN long-reach interfaces

Amendment 1

ITU-T G-SERIES RECOMMENDATIONS

TRANSMISSION SYSTEMS AND MEDIA, DIGITAL SYSTEMS AND NETWORKS

INTERNATIONAL TELEPHONE CONNECTIONS AND CIRCUITS	G.100–G.199
GENERAL CHARACTERISTICS COMMON TO ALL ANALOGUE CARRIER-TRANSMISSION SYSTEMS	G.200–G.299
INDIVIDUAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON METALLIC LINES	G.300–G.399
GENERAL CHARACTERISTICS OF INTERNATIONAL CARRIER TELEPHONE SYSTEMS ON RADIO-RELAY OR SATELLITE LINKS AND INTERCONNECTION WITH METALLIC LINES	G.400–G.449
COORDINATION OF RADIOTELEPHONY AND LINE TELEPHONY	G.450–G.499
TRANSMISSION MEDIA AND OPTICAL SYSTEMS CHARACTERISTICS	G.600–G.699
DIGITAL TERMINAL EQUIPMENTS	G.700–G.799
General	**G.700–G.709**
Coding of voice and audio signals	G.710–G.729
Principal characteristics of primary multiplex equipment	G.730–G.739
Principal characteristics of second order multiplex equipment	G.740–G.749
Principal characteristics of higher order multiplex equipment	G.750–G.759
Principal characteristics of transcoder and digital multiplication equipment	G.760–G.769
Operations, administration and maintenance features of transmission equipment	G.770–G.779
Principal characteristics of multiplexing equipment for the synchronous digital hierarchy	G.780–G.789
Other terminal equipment	G.790–G.799
DIGITAL NETWORKS	G.800–G.899
DIGITAL SECTIONS AND DIGITAL LINE SYSTEM	G.900–G.999
MULTIMEDIA QUALITY OF SERVICE AND PERFORMANCE – GENERIC AND USER-RELATED ASPECTS	G.1000–G.1999
TRANSMISSION MEDIA CHARACTERISTICS	G.6000–G.6999
DATA OVER TRANSPORT – GENERIC ASPECTS	G.7000–G.7999
PACKET OVER TRANSPORT ASPECTS	G.8000–G.8999
ACCESS NETWORKS	G.9000–G.9999

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T G.709.3/Y.1331.3

Flexible OTN long-reach interfaces

Amendment 1

Summary

Recommendation ITU-T G.709.3/Y.1331.3 defines the flexible optical transport network (OTN), known as FlexO, long-reach interfaces that support bonding (i.e., grouping) of multiple of these interfaces such that one or more client signals (e.g., one or more OTUCn ($n \geq 1$)) can be transferred via one or more optical tributary signals (OTSi) over one or more physical interfaces. The Recommendation specifies the frame structure for FlexO long reach interfaces using forward error correction codes with a higher coding gain than used in the FlexO short reach interfaces that are specified in Recommendation ITU-T G.709.1/Y.1331.1 and multiplexing of OTUCn client signals into the payload of a FlexO group.

Edition 2 contains the following extensions to Edition 1.1:

– Addition of 100G, 200G and 400G FlexO with OFEC (16, Annexes D, E, G, Appendices III, IV, V, bibliography)

– Addition of 100G FlexO with concatenated FEC (15.4.1, 15.5.4)

– Addition of multiplexing of OTUCn client signals into the payload of a FlexO group (Annex F).

This Recommendation includes an electronic attachment with the worksheets specified in Appendix IV.

Amendment 1 updates the text in Annex G to support the FlexO-x-DO TS, PS and MFAS overhead bit values.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T G.709.3/Y.1331.3	2018-06-22	15	11.1002/1000/13523
1.1	ITU-T G.709.3/Y.1331.3 (2018) Amd. 1	2018-11-29	15	11.1002/1000/13788
2.0	ITU-T G.709.3/Y.1331.3	2020-12-22	15	11.1002/1000/14499
2.1	ITU-T G.709.3/Y.1331.3 (2020) Amd. 1	2022-11-13	15	11.1002/1000/15136

Keywords

C-FEC, FlexO, long reach, O-FEC, OTN, SC-FEC.

* To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents/software copyrights, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the appropriate ITU-T databases available via the ITU-T website at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Electronic attachment: Worksheets specified in Appendix IV.

Recommendation ITU-T G.709.3/Y.1331.3

Flexible OTN long-reach interfaces

Amendment 1

Editorial note: This is a complete-text publication. Modifications introduced by this amendment are shown in revision marks relative to Recommendation ITU-T G.709.3/Y.1331.3 (2020).

1 Scope

This Recommendation[1] defines the flexible optical transport network (OTN), known as FlexO, long-reach interfaces that support bonding (i.e., grouping) of multiple of these interfaces such that an OTUCn ($n \geq 1$) can be transferred via one or more optical tributary signals (OTSi) over one or more physical interfaces.

The optical parameters associated with FlexO long reach interfaces are provided by application codes, which are specified for 100G in [ITU-T G.698.2] and are expected to be specified for other rates in a future edition of [ITU-T G.698.2].

A FlexO long reach interface group complements the OTN functionality specified in [ITU-T G.709] and [ITU-T G.709.1] such as B100G OTUCn frame, ODUk/flex, optical interface bonding for short reach interfaces, FlexO interface group management and OTUCn (de)mapping into/from FlexO group payload area, with new functionalities supporting optical interface bonding for long reach interfaces and multiplexing of multiple $OTUCn_i$ into the payload of the FlexO group.

This Recommendation provides specifications for new functionalities that are specific to FlexO long reach interface groups and refers to [ITU-T G.709], [ITU-T G.709.1], [ITU-T G.709.2] and [ITU-T G.798] for already existing functionalities. In addition, some introduction material for the addressed applications is included.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T G.698.2] Recommendation ITU-T G.698.2 (2018), *Amplified multichannel dense wavelength division multiplexing applications with single channel optical interfaces.*

[ITU-T G.709] Recommendation ITU-T G.709/Y.1331 (2020), *Interfaces for the optical transport network.*

[ITU-T G.709.1] Recommendation ITU-T G.709.1/Y.1331.1 (2018), *Flexible OTN short-reach interface*.

---

[1] This Recommendation includes an electronic attachment with the worksheets specified in Appendix IV. The electronic attachment is only distributed with the text for the base Recommendation.

[ITU-T G.709.2] Recommendation ITU-T G.709.2/Y.1331.2 (2018), *OTU4 long-reach interface.*

[ITU-T G.798] Recommendation ITU-T G.798 (2017), *Characteristics of optical transport network hierarchy equipment functional blocks.*

[ITU-T G.872] Recommendation ITU-T G.872 (2019), *Architecture of optical transport networks.*

[ITU-T G.975.1] Recommendation ITU-T G.975.1 (2004), *Forward error correction for high bit-rate DWDM submarine systems*.

[IEEE 802.3] IEEE Std. 802.3-2018, *Standard for Ethernet.*

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 Terms defined in [ITU-T G.709]

– completely standardized OTUCn (OTUCn)

– optical data unit (ODUCn)

– optical payload unit (OPUCn)

– optical transport network (OTN)

– optical data unit k (ODUk)

– optical tributary signal assembly (OTSiA)

3.1.2 Terms defined in [ITU-T G.709.1]

– FlexO

– FlexO-x

– FlexO-x-RS

– FlexO-x-RS interface

– FlexO-x-RS-m interface group

– FOICx.k-RS

– FOICx.k-RS lane.

3.1.3 Terms defined in [ITU-T G.709.2]

– Base block

3.1.4 Terms defined in [ITU-T G.975.1]

– coding gain

– net coding gain.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 FEC Block Group (FBG): Refers to a group of 1,305,430 contiguous bits in consecutive FlexO-x-SC rows which maps to five base blocks.

3.2.2 FlexO-x-DO: Information structure consisting of a FlexO-x that is carried in the payload of a FlexO-x-D<fec> (DSP) frame with Open FEC parity and overhead.

3.2.3 FlexO-x-DO interface: Refers to an individual member interface that is part of a FlexO-x-DO-m interface group.

3.2.4 FlexO-x-DO-m interface group: Refers to the group of m × FlexO-x-DO interfaces.

NOTE – The text may use "FlexO group" as short-hand for FlexO-x-DO-m interface group.

3.2.5 FlexO-x-DSH: Information structure consisting of a FlexO-x that is carried in the payload of a FlexO-x-D<fec> (DSP) frame with staircase and hamming SD FEC parity and overhead.

3.2.6 FlexO-x-DSH interface: Refers to an individual member interface that is part of a FlexO-x-DSH-m interface group.

3.2.7 FlexO-x-DSH-m interface group: Refers to the group of m × FlexO-x-DSH interfaces.

NOTE – The text may use "FlexO group" as short-hand for FlexO-x-DSH-m interface group.

3.2.8 FlexO-x-SC: Information structure consisting of a FlexO-x plus staircase FEC parity and overhead.

3.2.9 FlexO-x-SC interface: Refers to an individual member interface that is part of a FlexO-x-SC-m interface group.

3.2.10 FlexO-x-SC-m interface group: Refers to the group of m × FlexO-x-SC interfaces.

NOTE – The text may use "FlexO group" as short-hand for FlexO-x-SC-m interface group.

3.2.11 FOICx.k-DO: Refers to a FlexO-x-DO interface using k parallel FOICx.k-DO lanes.

3.2.12 FOICx.k-DO lane: Refers to an electrical/optical lane of a FlexO-x-DO.

3.2.13 FOICx.k-DSH: Refers to a FlexO-x-DSH interface using k parallel FOICx.k-DSH lanes.

3.2.14 FOICx.k-DSH lane: Refers to an electrical/optical lane of a FlexO-x-DSH.

3.2.15 FOICx.k-SC: Refers to a FlexO-x-SC interface using k parallel FOICx.k-SC lanes.

NOTE – "FOICx.k" is the FlexO equivalent of "OTLk.m" for OTUk as defined in [ITU-T G.709].

3.2.16 FOICx.k-SC lane: Refers to an electrical/optical lane of a FlexO-x-SC.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

AM	Alignment Marker
B100G	Beyond 100G
BCH	Bose-Chaudhuri-Hocquengham
DWDM	Dense Wavelength Division Multiplex
ED	Error Decorrelator
EDI	Error Decorrelator Interleaver
EDD	Error Decorrelator De-interleaver
FAW	Fame Alignment Word
FBA	FEC Block Alignment
FBG	FEC Block Group
FEC	Forward Error Correction
FlexO	Flexible OTN
FlexO-x	FlexO of order x

FlexO-x-SC	FlexO of order x with staircase FEC
FlexO-x-DSH	FlexO of order x with DSP frame and staircase FEC plus Hamming SD FEC
FlexO-x-DO	FlexO of order x with DSP frame and open FEC
FOI	FlexO Interface
FOICx.k	FlexO Interface of order Cx with k lanes
LR	Long Reach
MBAS	Multi Block Alignment Signal
MFAS	Multi-Frame Alignment Signal
MLD	Multi-Lane Distribution
NCG	Net Coding Gain
NNI	Network Node Interface
OCC	Overhead Communication Channel
ODU	Optical Data Unit
ODU*k*	ODU order *k*
ODU*Cn*	ODU order *Cn*
OFBG	Open FEC Block Group
OFBGP	Open FEC Block Group with Parity
OFBGPz	Open FEC Block Group with Parity, order z
OFBGz	Open FEC Block Group, order z
OFC	Open FEC Coder
OFCi	Open FEC Coder, part i (i = 0,1)
OFCP	Open FEC Coder with Parity bits
OFCPi	Open FEC Coder with Parity bits, part i (i = 0,1)
OFEC	Open FEC
OSMC	OTN Synchronization Messaging Channel
OTN	Optical Transport Network
OTSi	Optical Tributary Signal
OTU	Optical Transport Unit
OTU*Cn*	OTU order *Cn*
OTUCn_i	i^{th} OTU*Cn* instance in a set of OTUCn multiplexed into the payload of a FlexO group
SC FEC	Staircase FEC
RS	Reed Solomon

5 Conventions

This Recommendation uses the following conventions:

k: The index "k" is used to represent a supported bit rate and the different versions of OPUk, ODUk and OTUk. Example for k are

	"1" for an approximate bit rate of 2.5 Gbit/s, "2" for an approximate bit rate of 10 Gbit/s, and "3" for an approximate bit rate of 40 Gbit/s.
Cn:	The index Cn is used for n × 100G (C = 100G).
Z:	The index "Z" is used to represent the number of bits per dual polarization symbol or block and column in the FlexO-x-D<fec> frame structure. Z=4 corresponds to DP-QPSK and Z=8 corresponds to DP-16QAM.
Transmission order:	The order of transmission of information in all the diagrams in this Recommendation is first from left to right and then from top to bottom. Within each byte the most significant bit is transmitted first. The most significant bit (bit 1) is illustrated at the left in all the diagrams.
Value of reserved bit(s):	The value of an overhead bit, which is reserved or reserved for future international standardization, shall be set to "0".
Value of non-sourced bit(s):	Unless stated otherwise, any non-sourced bits shall be set to "0".
⌊ ⌋:	Denotes the floor operator,
(a % b):	Denotes the value of a modulo b,
(a ^ b):	Represents the number with a binary representation equal to the bit-wise "exclusive or" of the binary representations of the numbers a and b.

6 Introduction and applications

The FlexO long reach interface groups specified in this Recommendation provide a longer reach version of the FlexO short reach interface groups specified in [ITU-T G.709.1]. In order to mitigate the impairments of accumulated noise it uses forward error correction (FEC) types with a higher coding gain than the FEC type deployed in FlexO short reach interfaces.

Example applications are shown in Appendix I.

This Recommendation specifies FlexO-x-<fec>-m interface groups, of which x ≥ 1, m ≥ 1 and <fec> represents a FEC with a net coding gain higher than the RS(544,514) FEC in FlexO-x-RS-m interface groups specified in [ITU-T G.709.1]. It also specifies multiplexing of multiple OTUCn signals into the payload of FlexO Group.

NOTE – The interfaces specified in this Recommendation are not envisaged to be reused on a system internal interface (i.e., a module framer interface), instead such interfaces can deploy the FOI*Cx.k* as specified in [b-ITU-T G-Sup.58].

7 Structure and processes

This clause introduces the functions associated with a FlexO-x-SC-m, FlexO-x-DSH-m and FlexO-x-DO-m interface group and the basic signal structure, processes and atomic functions.

7.1 FlexO-x-SC-m signal structure

The FlexO-x-SC-m interface group in this Recommendation is only specified for long-reach applications. For short reach applications, refer to [ITU-T G.709.1]. The FlexO-x-SC-m interface group functional model is specified in [ITU-T G.872].

NOTE – The physical optical interface specifications are beyond the scope of this Recommendation.

The information structure for FlexO-x-SC-m interface groups is represented by information containment relationships and information flows. The principal information containment relationship is shown in Figure 7-1.

One OTUCn signal (consisting of n OTUC instances) is mapped into the payload of n FlexO signals, each FlexO signal containing the bits of one OTUC signal. The n FlexO signals are mapped into m ($m = \lceil n/x \rceil$) FlexO-x-SC signals, each FlexO-x-SC signal containing "x" (frame/multi-frame aligned interleaved) FlexO signals ($x \geq 1$) plus FEC parity and FlexO-x-SC overhead. Each FlexO-x-SC signal is split into k FlexO-x-SC lane signals (FOI*Cx.k*-SC). The k lane signals are modulated onto one OTSi, which is transported via one media element.

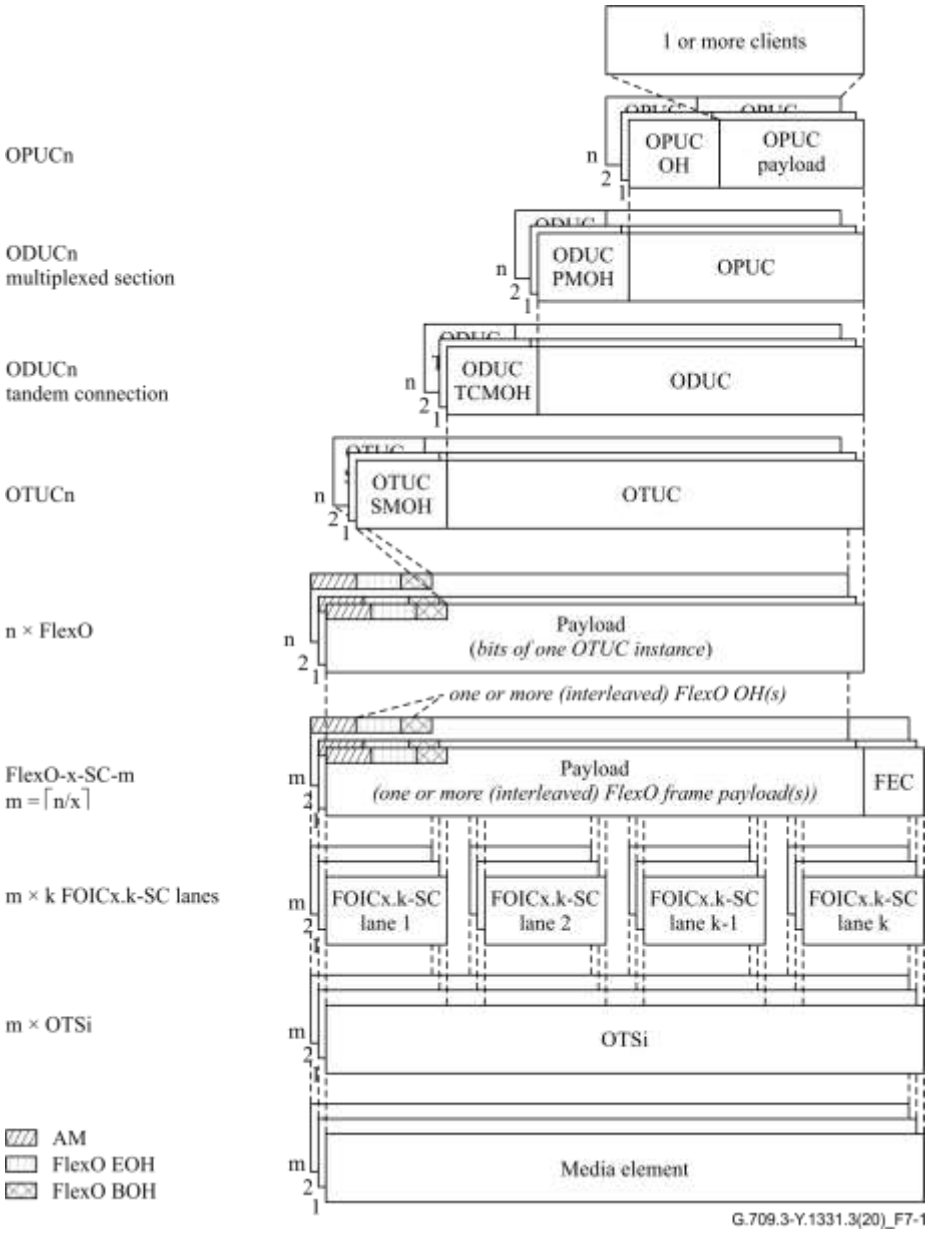


Figure 7-1 – FlexO-x-SC-m interface group principal information containment relationship

7.2 FlexO-x-DSH-m signal structure

The information structure for FlexO-x-DSH groups is represented by information containment relationships and flows. The principal information containment relationship is described in Figure 7-2.

One OTUCn signal (consisting of n OTUC instances) is mapped into n FlexO instances, each FlexO instance containing one OTUC instance. The n FlexO instances are mapped into one FlexO-x-SC-m signal (consisting of m = ⌈n/x⌉ FlexO-x-SC instances), each FlexO-x-SC instance containing "x" interleaved FlexO signals plus SC FEC parity.

The FlexO-x-SC-m signal is mapped into one FlexO-x-DSH-m signal (consisting of m FlexO-x-DSH instances, each FlexO-x-DSH instance containing one FlexO-x-SC signal plus Hamming SD FEC parity and overhead.

Each FlexO-x-DSH signal is split into k FOICx.k-DSH lane signals.

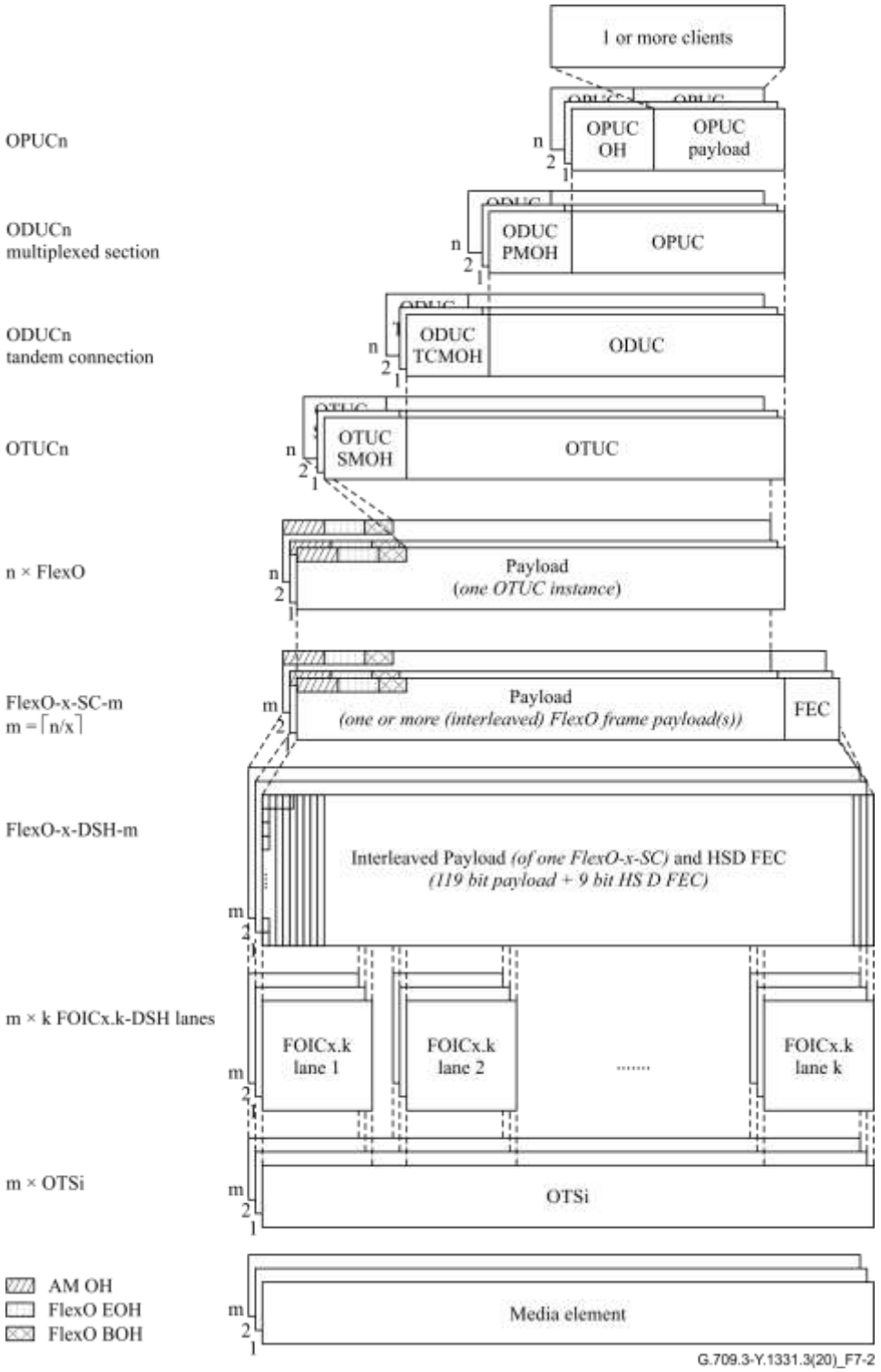


Figure 7-2 – FlexO-x-DSH group principal information containment relationship

7.3 FlexO-x-DO-m signal structure

The information structure for FlexO-x-DO groups is represented by information containment relationships and flows. The principal information containment relationship is described in Figure 7-3.

One OTUCn signal (consisting of n OTUC instances) is mapped into n FlexO instances, each FlexO instance containing one OTUC instance. The n FlexO instances are mapped into one FlexO-x-DO-m signal, consisting of m = ⌈n/x⌉ FlexO-x-DO instances. Each FlexO-x-DO instance contains one padded and scrambled FlexO-x signal consisting of "x" interleaved FlexO signals, plus BCH(256,239) SD FEC parity.

Each FlexO-x-DO signal is split into k FOICx.k-DO lane signals.

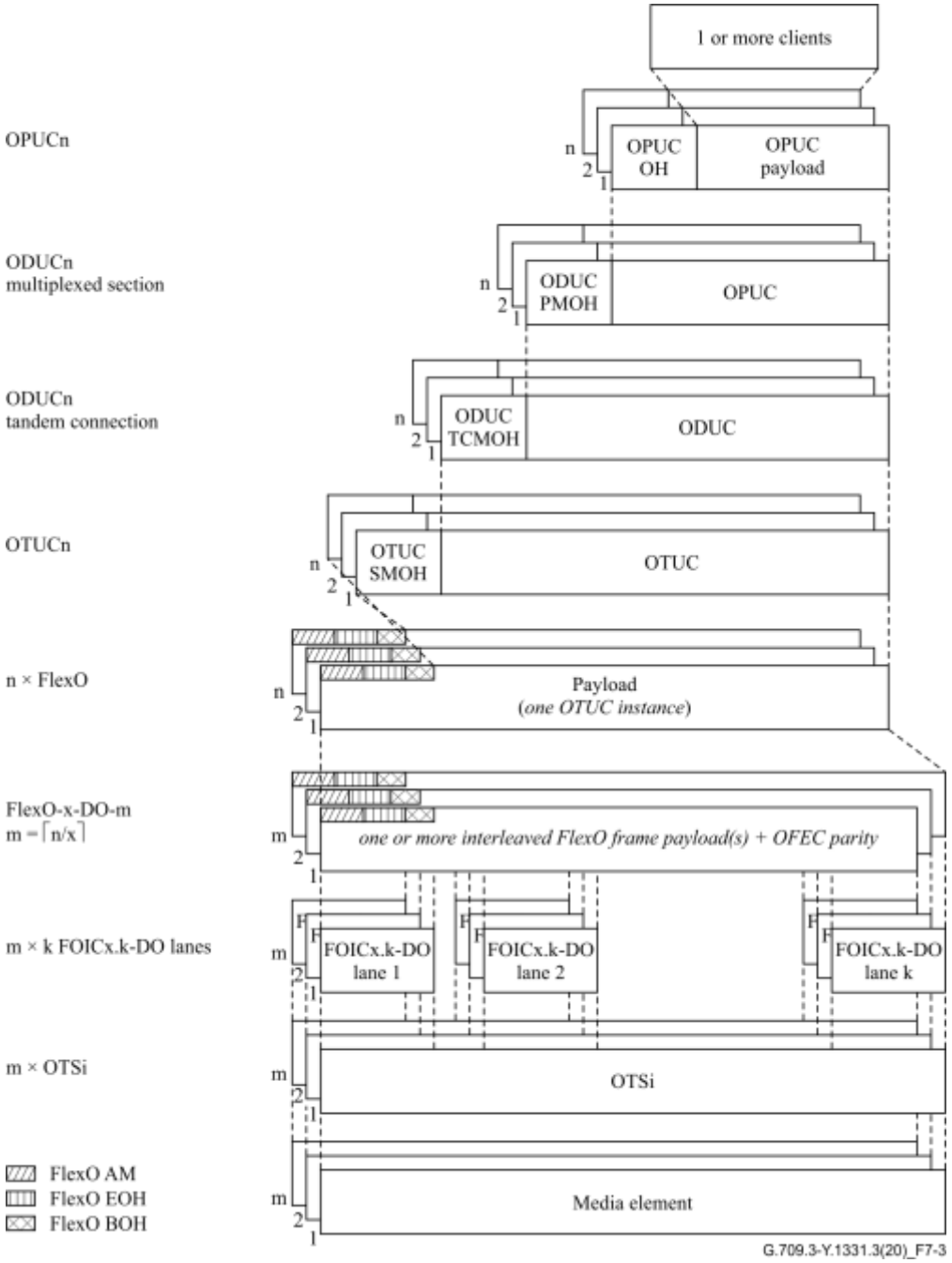


Figure 7-3 – FlexO-x-DO group principal information containment relationship

7.4 Processing and information flow

Functions and information flows are specified in [ITU-T G.798].

8 FlexO frame

Refer to clause 8 of [ITU-T G.709.1].

9 Alignment markers, PAD and FlexO overhead

Refer to clause 9 of [ITU-T G.709.1].

10 Mapping of OTUCn signal into n FlexO instances

Refer to clause 10 of [ITU-T G.709.1].

Deskewing in the sink process is performed between OTUC instances within the OTUCn as specified in [ITU-T G.709.1].

The skew requirements are intended to account for variations due to digital mapping, cable lengths and frequency slot related delay differences for relevant applications as defined by [ITU-T G.698.2]. The skew tolerance requirement for OTUC instances within an OTUCn that is carried over longer reach FlexO interface groups is 1 µs.

NOTE – This value of 1 µs is larger than the 300 ns value specified in [ITU-T G.709.1] for short reach interface groups.

11 m×100G FlexO with staircase FEC interface group (FlexO-1-SC-m)

A FlexO-1-SC-m interface group consists of m FlexO-1-SC interfaces.

11.1 FlexO-1-SC frame structure

The 100G FlexO-1-SC frame structure is shown in Figure 11-1 and consists of 128 rows by 5,485 1-bit columns. It contains one FlexO frame structure as defined in clause 8, extended with a SC FEC parity area in columns 5141 to 5485 in every row.

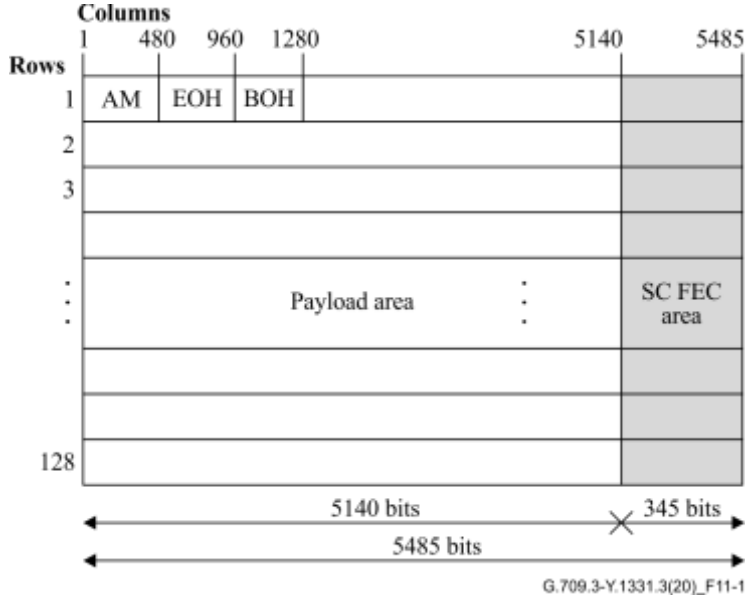


Figure 11-1 – 100G FlexO-1-SC frame structure

To accommodate the block length of the staircase FEC, which is not aligned with the 100G FlexO-1-SC frame length, an additional SC FEC block group (FBG) structure is superimposed on the underlying FlexO-1-SC frame structure (see Figure 11-2, left).

The FEC block group (FBG) consists of 238 consecutive FlexO-1-SC frame rows (of 5485-bit each), which map to five contiguous base blocks. The FBG contains 5×244,664 information and 5×16384 parity bits plus 5×38 FlexO-1-SC OH bits that are located between the parity bits of successive SC FEC blocks. A FBG contains the information bits of five SC FEC blocks B_i to B_{i+4} and parity bits of five SC FEC blocks B_{i-1} to B_{i+3}.

The boundaries of these five SC FEC information, parity and FlexO-1-SC overhead blocks within a FBG are defined in Table 11-1 and illustrated in the right segment of Figure 11-2. The start of the FBG is identified through a SC FEC block row number indication carried in the FBA field of the FlexO-1-SC overhead, part 1 (see Figure 11-3).

Table 11-1 – SC FEC information, parity and FlexO-1-SC part 2 overhead block boundaries

	Information blocks {row,column}	**Parity blocks {row,column}**	**FlexO-1-SC part 2 OH {row,column}**
1st block	From {j+1,1} to {j+48,3084}	From {j+1,5141} to {j+48,5309}	From {j+48,5310} to {j+48,5347}
2nd block	From {j+48,3085} to {1+96,1028}	From {j+48,5348} to {1+96,5171}	From {j+96,5172} to {i+96,5209}
3rd block	From {j+96,1029} to {j+143,4112}	From {j+96,5210} to {j+143,5378}	From {j+143,5379} to {j+143,5416}
4th block	From {j+143,4113} to {j+191,2056}	From {j+143,5417} to {j+191,5240}	From {j+191,5241} to {j+191,5278}
5th block	From {j+191,2057} to {j+238,5140}	From {j+191,5279} to {j+238,5447}	From j+191,5448} to {j+238,5485}

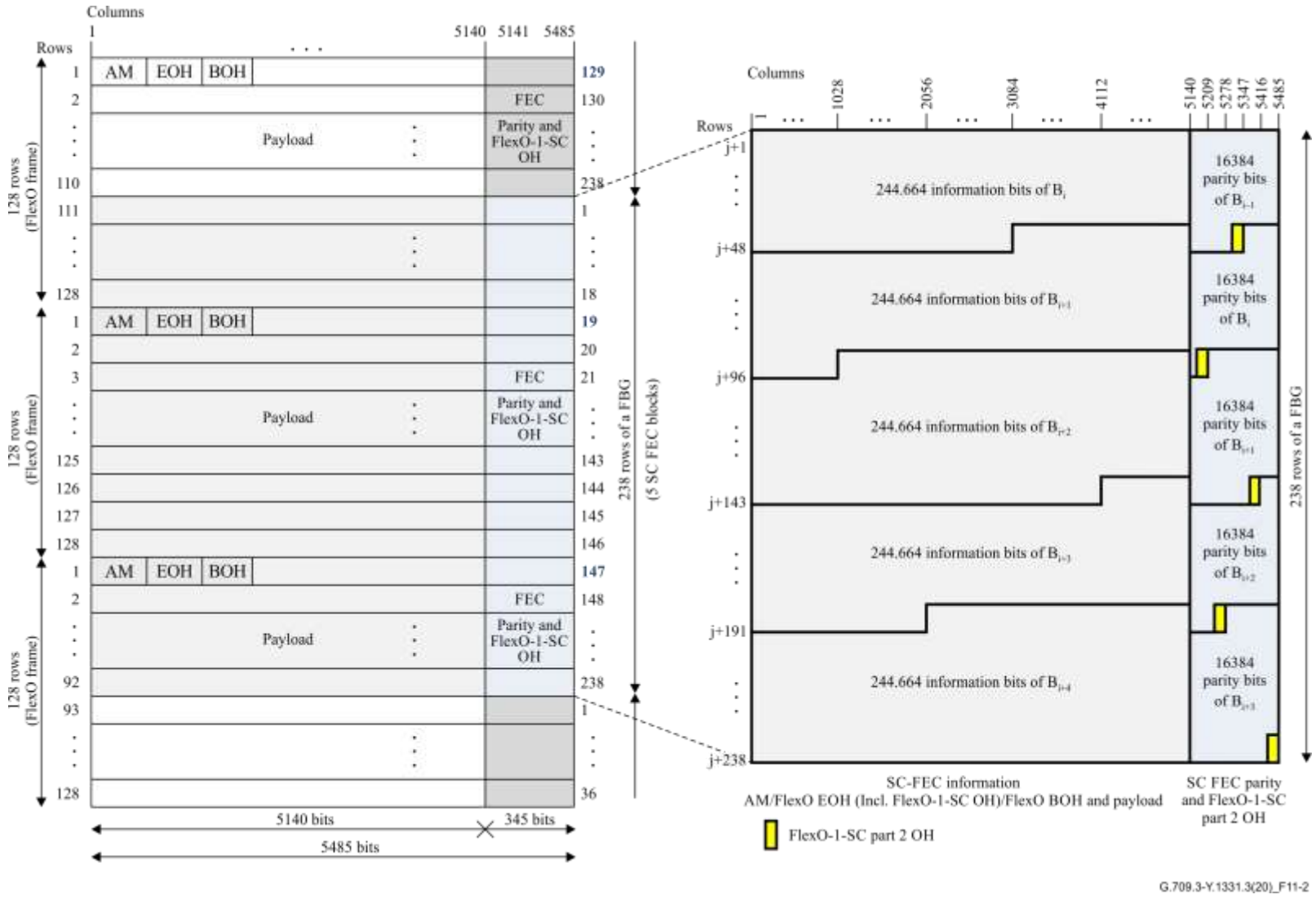


Figure 11-2 – 100G FlexO-1-SC's FBG structure (left) and information and parity block and FlexO-1-SC OH boundaries (right)

11.2 FlexO-1-SC bit rate and frame periods

The bit rate and tolerance of the 100G FlexO-1-SC signal is defined in Table 11-2.

Table 11-2 – FlexO-1-SC types and bit rates

100G FlexO-1-SC nominal bit rate	Bit-rate tolerance
524366/462961 × 99 532 800 kbit/s	±20 ppm
NOTE 1 – The nominal FlexO-1-SC bit rate is approximately: 112 734 368.996 kbit/s. NOTE 2 – The FlexO-1-SC bit rate can be based on the OTUC bit rate as follows: 4388/4097 × OTUC bit rate = 4388/4097 × 239/226 × 99 532 800 kbit/s. NOTE 3 – The FlexO-1-SC bit rate can be based on the FlexO-1-RS bit rate as follows: 1097/1088 × FlexO-1-RS bit rate = 1097/1088 × 256/241 × 239/226 × 99 532 800 kbit/s.	

The frame and multi-frame periods of the FlexO-1-SC signal are defined in Table 11-3.

Table 11-3 – FlexO-1-SC frame and multi-frame periods

Frame period (Note)	Multi-frame period (Note)
~6.228 µs	49.822 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.	

11.3 FlexO-1-SC overhead

The FlexO-1-SC frame contains two overhead areas. The first overhead area is located in the first 8 bits of the FlexO EOH area in row 1, columns 481 to 488. It includes information to support the FBG alignment function. The second overhead area is located in the FlexO-1-SC FEC parity area, in blocks of 38 bits each between two successive SC FEC blocks.

The FlexO-1-SC OH is terminated where the FlexO-1-SC frame is assembled and disassembled.

An overview of the first part of the FlexO-1-SC OH area is presented in Figure 11-3. An overview of the second part of the FlexO-1-SC OH area is presented in Figure 11-2 and Figure 11-4.

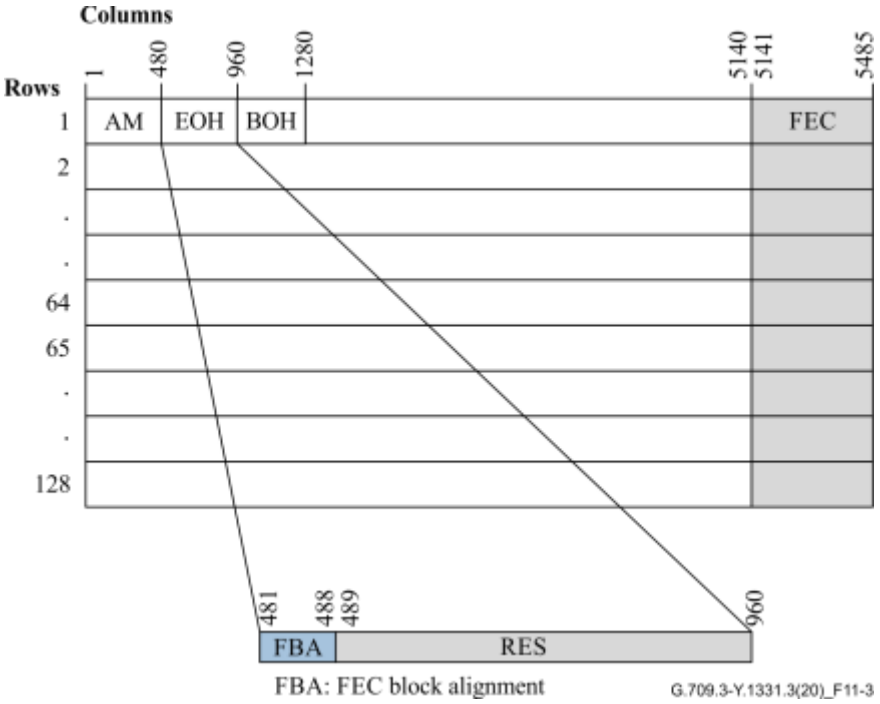


Figure 11-3 – FlexO-1-SC overhead, part 1

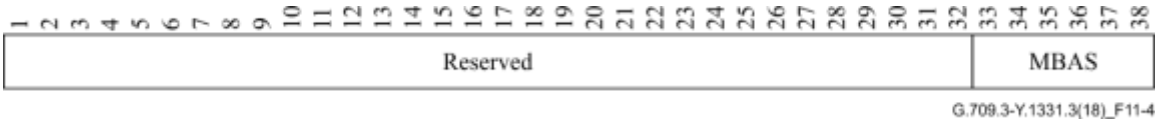


Figure 11-4 – FlexO-x-SC (x = 1,2,4) overhead, part 2

11.3.1 FlexO-1-SC FEC block alignment (FBA)

Figure 11-3 shows the "FBA" overhead byte in byte 1 of the first part of the FlexO-1-SC overhead area. The FEC block alignment (FBA) carries an 8-bit value indicating the current row number in the range 1 (0b0000 0000) to 238 (0b1110 1101) within the 238 FlexO-1-SC rows sequence of a FBG. Refer to Figure 11-5 for the encoding of the FBA values.

NOTE – FBA is not used in FlexO-1-DSH interfaces.

This row number value is used in the far end decoder to synchronize a 238 row counter that indicates the start of each FBG.

FBA = 1 (encoded as 0b0000 0000) means that the next start of a FBG is in this FlexO-1-SC frame row. FBA > 1 means that the next start of a FBG is (238 – FBA + 1) FlexO-1-SC frame rows later.

Successive FBA values will have the following relationship:

$$FBA(t) = 1 + (FBA(t-1) - 1 + 128) \bmod 238.$$

The FBA sequence repeats every 119 FlexO-1-SC frames and is illustrated in Table 11-4. The FBA sequence contains the FBA values *1,129, 19, 147, 37, 165* at the start of the sequence and FBA values *93, 221, 111* at the end of the sequence. An example illustrating FBA values of 129, 19 and 147 is presented in Figure 11-6.

NOTE – The FBA sequence repeats every 119 FlexO-1-SC frames and includes only odd numbers, not every 238 frames.

Table 11-4 – FlexO-1-SC FBA sequence (top-to-bottom, left-to-right order)

1	181	123	65	7	187
129	71	13	193	135	77
19	199	141	83	25	205
147	89	31	211	153	95
37	217	159	101	43	223
165	107	49	229	171	113
55	235	177	119	61	3
183	125	67	9	189	131
73	15	195	137	79	21
201	143	85	27	207	149
91	33	213	155	97	39
219	161	103	45	225	167
109	51	231	173	115	57
237	179	121	63	5	185
127	69	11	191	133	75
17	197	139	81	23	203
145	87	29	209	151	93
35	215	157	99	41	221
163	105	47	227	169	111
53	233	175	117	59	

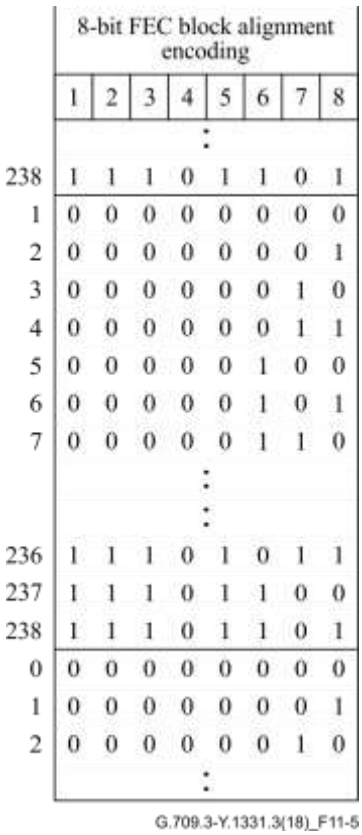


Figure 11-5 – FlexO-1-SC FEC block alignment (FBA) signal overhead

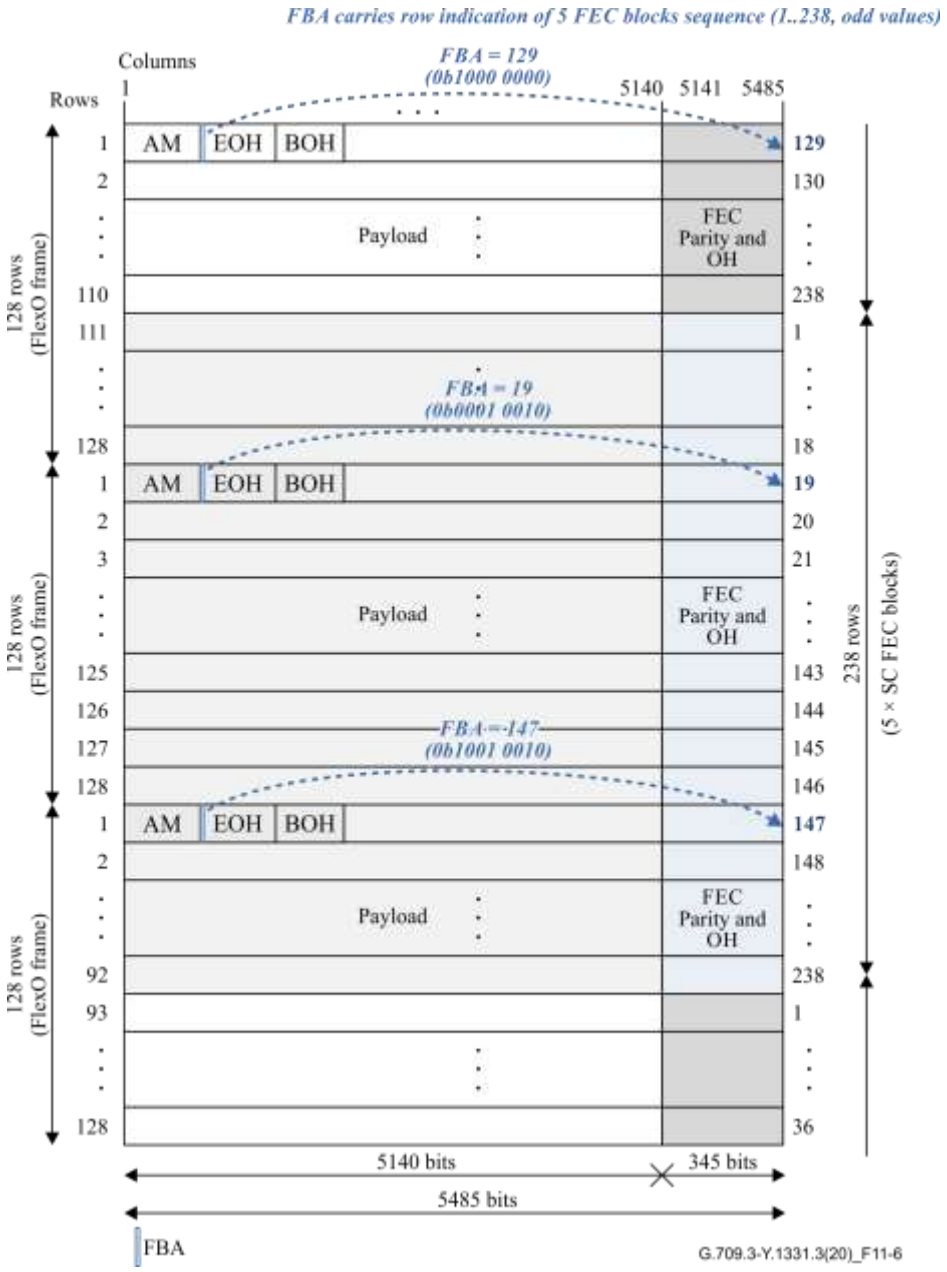


Figure 11-6 – FlexO-1-SC FEC block alignment (FBA) signal operation example

11.3.2 Multi block alignment signal (MBAS)

To synchronize the state of the Error Decorrelator (ED) controllers between the receiver and the transmitter, the staircase FEC scheme uses a 7-bit SC FEC multi block alignment signal (MBAS) which provides a 128 block sequence.

The six most significant bits of the 7-bit MBAS are transferred between source and sink in the 6-bit MBAS overhead, which is located in bits 33 to 38 of the second part of the FlexO-x-SC OH area (x = 1,2,4).

The numerical value represented in the six MBAS overhead bits will be incremented every two SC FEC blocks and provides as such a 128-block multi-block as illustrated in Figure 11-7.

	7-bit multi-block alignment signal						
	1	2	3	4	5	6	7
	6-bit MBAS OH						
	33	34	35	36	37	38	
				⋮			
0	0	0	0	0	0	0	*0*
1	0	0	0	0	0	0	*1*
2	0	0	0	0	0	1	*0*
3	0	0	0	0	0	1	*1*
4	0	0	0	0	1	0	*0*
5	0	0	0	0	1	0	*1*
6	0	0	0	0	1	1	*0*
7	0	0	0	0	1	1	*1*
				⋮			⋮
				⋮			⋮
125	1	1	1	1	1	0	*1*
126	1	1	1	1	1	1	*0*
127	1	1	1	1	1	1	*1*
0	0	0	0	0	0	0	*0*
1	0	0	0	0	0	0	*1*
2	0	0	0	0	0	1	*0*
				⋮			⋮

G.709.3-Y.1331.3(18)_F11-7

Figure 11-7 – Multi-block alignment signal overhead

11.4 Staircase forward error correction (SC FEC)

The FlexO-1-SC FEC code is a 512-bit × 510-bit generalized staircase code (SC FEC) that works in conjunction with an error decorrelator. The error decorrelator function is used to randomize the error locations, in order to reduce impact of correlated errors on the decoder performance of the random error correcting SC FEC. The SC FEC code is systematic and the 6.7% FlexO-1-SC FEC area specified in clause 11.1 is used to store the parity information generated by the encoder.

The operation of a generic staircase FEC scheme (with error de-correlator) is specified in Annex A. The FlexO-1-SC specific aspects of the staircase FEC operation are specified in Annex B.

11.5 FlexO-1-SC scrambling

The FlexO-1-SC signal must have sufficient bit timing content at the network node interface (NNI). A suitable bit pattern, which prevents a long sequence of "1"s or "0"s, is provided by using an additive scrambler.

Scrambling of the FlexO-1-SC signal is performed after SC FEC parity computation and insertion into the FEC parity area of the FlexO-1-SC signal.

The operation of the scrambler shall be functionally equivalent to that of a frame-synchronous additive scrambler with a sequence length of 65535 and the generating polynomial shall be $x^{16} + x^{12} + x^3 + x + 1$. See Figure 11-3 [ITU-T G.709] for an illustration of this scrambler.

The scrambler state resets to 0xFFFF on the bit in row 1, column 481 and the scrambler state advances with each consecutive bit of the FlexO-1-SC frame (Figure 11-1). The bit in row 1, column 481 and all subsequent bits to be scrambled shall be added modulo 2 to the output from the x^{16} position of the scrambler. The scrambler shall run continuously throughout the complete

FlexO-1-SC frame (including the FEC parity area). The alignment markers (AM) bits shall not be scrambled. See Figure 11-8.

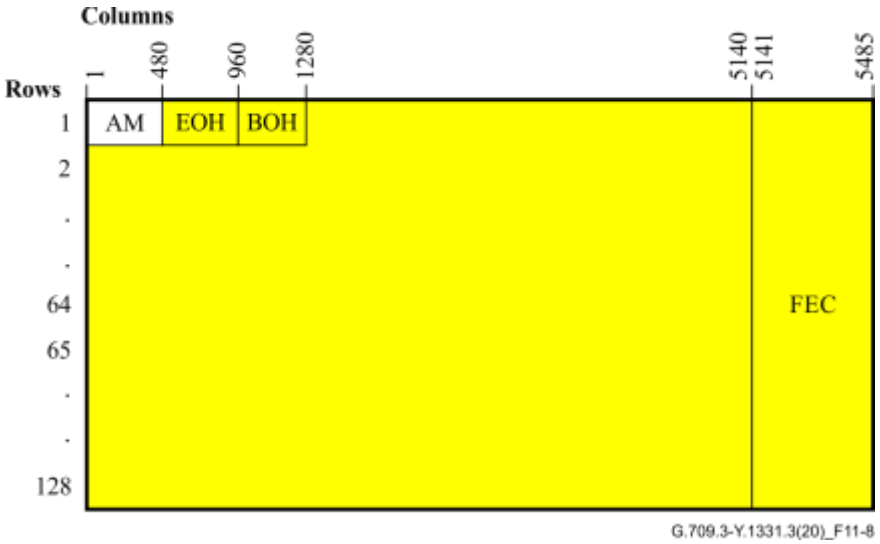


Figure 11-8 – FlexO-1-SC scrambling area

11.6 FOIC1.k-SC

A conceptually serial FlexO-1-SC signal is adapted to a parallel multi-lane distribution (MLD) signal format with k lanes, referred to as FOIC1.k-SC.

11.6.1 FOIC1.4-SC lanes

The FlexO-1-SC bits are distributed to four logical FOIC1.k-SC lanes, in groups of 10-bits, in a round robin distribution scheme from the lowest to the highest numbered lanes. The FOIC1.4-SC interface includes the 10-bit distribution, reordering and deskewing functions following the principles described in clause 11 of [ITU-T G.709.1]. Each FOIC1.4-SC lane is synchronous to the FlexO-1-SC frame.

Each FlexO-1-SC frame contains 5485 × 128 = 702080 bits. Each FOIC1.4-SC lane will carry 25% of these bits, which are 175520 bits, or 17552 10-bit blocks.

The resulting per-lane transmitted values of the AM fields are illustrated in Table 11-3 of [ITU-T G.709.1].

The FOIC1.4-SC interface format is intended for applications defined by application codes for metro networks in [ITU-T G.698.2]. Mapping of the four lanes to and the specification of the optical tributary signal (OTSi) is defined in [ITU-T G.698.2].

11.6.2 FOIC1.4-SC lane skew tolerance requirements

The tolerated skew between lanes in a FOIC1.4-SC interface signal is at least 180 ns.

11.6.3 FOIC1.4-SC 28G lane bit rate

The bit rates and tolerance of the FOIC1.4-SC lanes are defined in Table 11-5.

Table 11-5 – FOIC1.4-SC lane bit rates

FOIC1.4-SC lane nominal bit rate	Bit-rate tolerance
524366/462961 × 24 883 200 kbit/s	±20 ppm
NOTE – The nominal FOIC1.4-SC lane bit rates are approximately: 28 183 592.249 kbit/s.	

12 200G FlexO with staircase FEC frame structure (FlexO-2-SC)

12.1 FlexO-2-SC frame structure

The 200G FlexO-2-SC frame structure is shown in Figure 12-1 and consists of 128 rows by 10970 1-bit columns. It contains in columns 1 to 10280 a FlexO-2 as defined in clause 8, extended with a SC FEC parity area in columns 10281 to 10970.

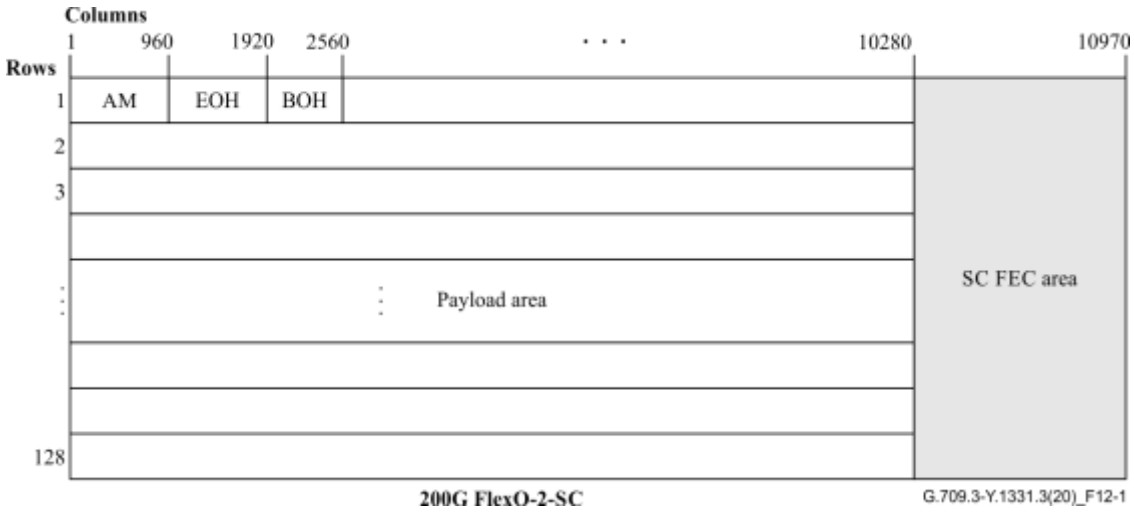


Figure 12-1 – 200G FlexO-2-SC frame structure

To accommodate the block length of the staircase FEC, which is not aligned with the 200G FlexO-2-SC frame length, an additional SC FEC block group (FBG) structure is superimposed on the underlying FlexO-2-SC frame structure (see Figure 12-2, left).

The FBG consists of 119 consecutive FlexO-2-SC frame rows (of 10970-bit each), which map to five contiguous base blocks. The FBG contains 5×244,664 information and 5×16384 parity bits plus 5×38 FlexO-2-SC OH bits that are located between the parity bits of successive SC FEC blocks. A FBG contains the information bits of five SC FEC blocks B_i to B_{i+4} and parity bits of five SC FEC blocks B_{i-1} to B_{i+3}.

The boundaries of these five SC FEC information, parity and FlexO-2-SC overhead blocks within a FBG are defined in Table 12-1 and illustrated in the right segment of Figure 12-2. The start of the FBG is identified through a SC FEC block row number indication carried in the FBA field of the FlexO-2-SC overhead, part 1 (see Figure 12-3).

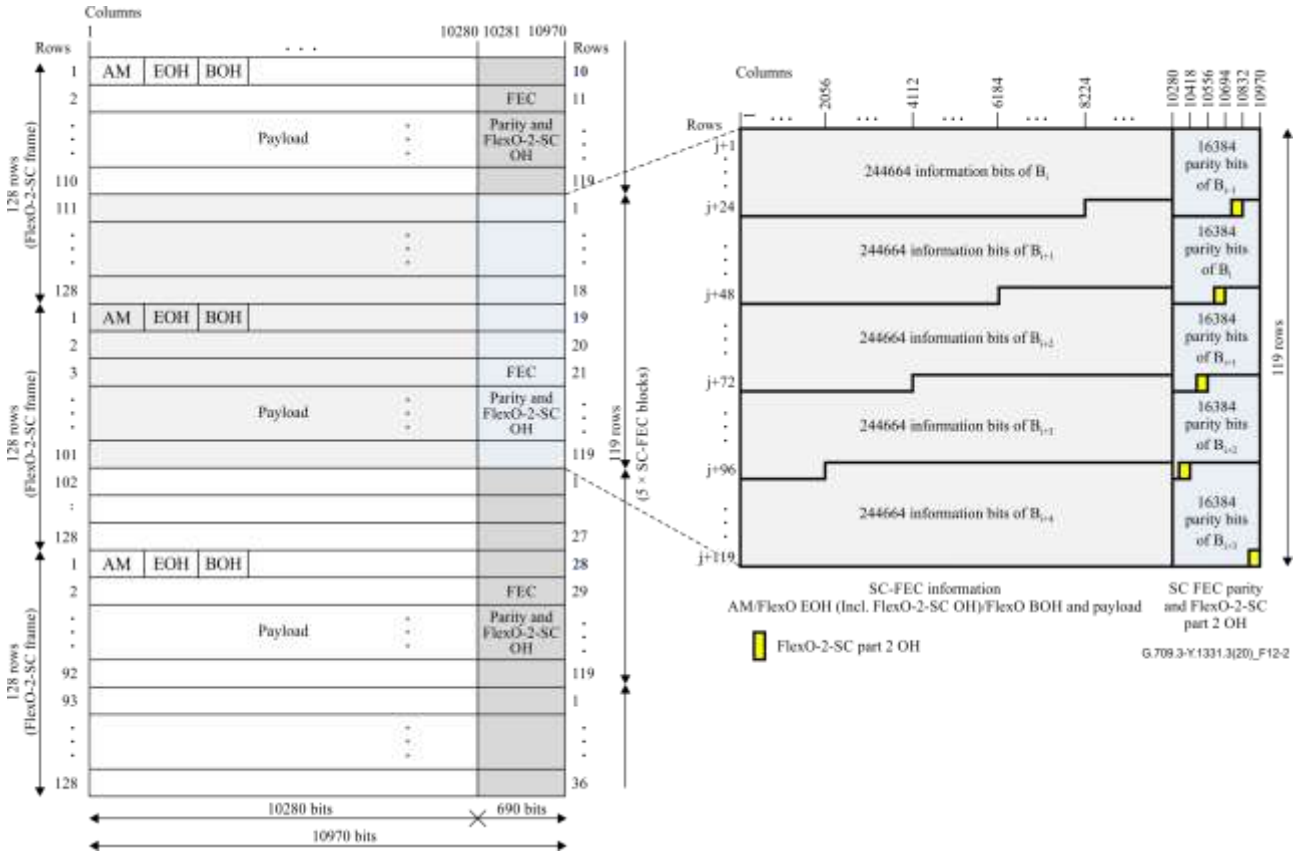


Figure 12-2 – FlexO-2-SC's FBG structure (left) and information and parity block and FlexO-2-SC OH boundaries (right)

Table 12-1 – SC FEC information, parity and FlexO-x-SC (x = 2,4) part 2 overhead block boundaries

	Information blocks {row,column}	**Parity blocks {row,column}**	**FlexO-2-SC part 2 OH {row,column}**
1st block	From {j+1,1} to {j+24,8224}	From {j+1,10281} to {j+24,10794}	From {j+24,10795} to {j+24,10832}
2nd block	From {j+24,8225} to {1+48,6184}	From {j+24,10833} to {1+48,10656}	From {j+48,10657} to {i+48,10694}
3rd block	From {j+48,6185} to {j+72,4112}	From {j+48,10695} to {j+72,10518}	From {j+72,10519} to {j+72,10556}
4th block	From {j+72,4113} to {j+96,2056}	From {j+72,10557} to {j+96,10380}	From {j+96,10381} to {j+96,10418}
5th block	From j+96,2057} to {j+119,10280}	From j+96,10419} to {j+119,10932}	From j+96,10933} to {j+119,10970}

12.2 FlexO-2-SC bit rate and frame periods

The bit rate and tolerance of the FlexO-2-SC signal is defined in Table 12-2.

Table 12-2 – FlexO-2-SC types and bit rates

FlexO-2-SC nominal bit rate	Bit-rate tolerance
524366/462961 × 2 × 99 532 800 kbit/s	±20 ppm
NOTE 1 – The nominal FlexO-2-SC bit rate is approximately: 225 468 737,992 kbit/s. NOTE 2 – The FlexO-2-SC bit rate can be based on the OTUC bit rate as follows: 4388/4097 × 2 × OTUC bit rate = 4388/4097 × 2 × 239/226 × 99 532 800 kbit/s. NOTE 3 – The FlexO-2-SC bit rate can be based on the FlexO-2-RS bit rate as follows: 1097/1088 × FlexO-2-RS bit rate = 1097/1088 × 256/241 × 2 × 239/226 × 99 532 800 kbit/s.	

The frame and multi-frame periods of the FlexO-2-SC signal are defined in Table 12-3.

Table 12-3 – FlexO-2-SC frame and multi-frame periods

Frame period (Note)	Multi-frame period (Note)
~6.228 µs	49.822 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.	

12.3 FlexO-2-SC overhead

The FlexO-2-SC frame contains two overhead areas. The first overhead area is located in the first 8 bits of the FlexO EOH in row 1, columns 961 to 968. It includes information to support the FBG alignment function. The second overhead area is located in the FlexO-2-SC FEC parity area, in blocks of 38 bits each between two successive SC FEC blocks.

The FlexO-2-SC OH is terminated where the FlexO-2-SC frame is assembled and disassembled.

An overview of the first part of the FlexO-2-SC OH area is presented in Figure 12-3. An overview of the second part of the FlexO FEC OH area is presented in Figure 12-2 and Figure 11-4.

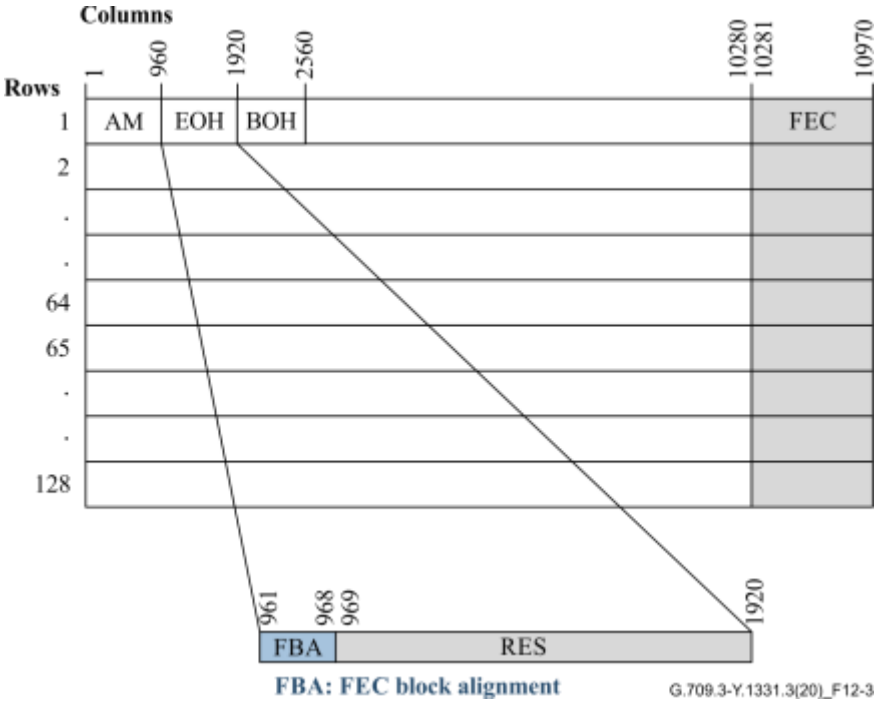


Figure 12-3 – FlexO-2-SC overhead, part 1

12.3.1 FlexO-2-SC FEC block alignment (FBA)

Reserved for future use. Not used in FlexO-2-DSH interfaces.

12.3.2 Multi block alignment signal (MBAS)

Refer to clause 11.3.2.

12.4 Staircase forward error correction (SC FEC)

The FlexO-2-SC FEC code is a 512-bit × 510-bit generalized staircase code (SC FEC) that works in conjunction with an error de-correlator. The error de-correlator function is used to randomize the error locations, in order to reduce impact of correlated errors on the decoder performance of the random error correcting SC FEC. The SC FEC code is systematic, and the 6.7% FlexO-2-SC FEC area specified in clause 12.1 is used to store the parity information generated by the encoder.

The generic operation of the staircase FEC scheme (with error de-correlator) is specified in Annex A. The FlexO-2-SC specific aspects of the staircase FEC operation are specified in Annex C.

13 400G FlexO with staircase FEC frame structure (FlexO-4-SC-m)

13.1 FlexO-4-SC frame structure

The 400G FlexO-4-SC frame structure is shown in Figure 13-1 and consists of 256 rows by 10970 1-bit columns. It contains in columns 1 to 10280 a FlexO-4 as defined in clause 8, extended with a SC FEC parity area in columns 10281 to 10970.

To accommodate the block length of the staircase FEC, which is not aligned with the 400G FlexO-4-SC frame length, an additional SC FEC block group (FBG) structure is superimposed on the underlying FlexO-4-SC frame structure (see Figure 13-2, left).

The FBG consists of 119 consecutive FlexO-4-SC frame rows (of 10970-bit each), which map to five contiguous base blocks. The FBG contains 5×244,664 information and 5×16384 parity bits plus 5×38 FlexO-4-SC OH bits that are located between the parity bits of successive SC FEC blocks. A FBG contains the information bits of five SC FEC blocks B_i to B_{i+4} and parity bits of five SC FEC blocks B_{i-1} to B_{i+3}.

The boundaries of these five SC FEC information, parity and FlexO-4-SC overhead blocks within a FBG are defined in Table 12-1 and illustrated in the right segment of Figure 13-2. The start of the FBG is identified through a SC FEC block row number indication carried in the FBA field of the FlexO-4-SC overhead, part 1 (see Figure 13-3).

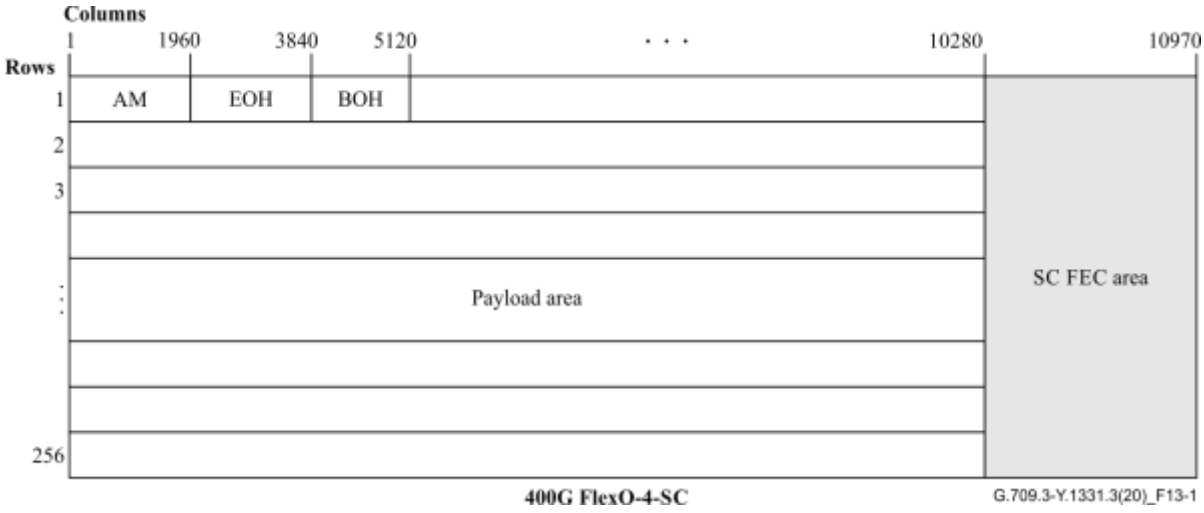


Figure 13-1 – 400G FlexO-4-SC frame structure

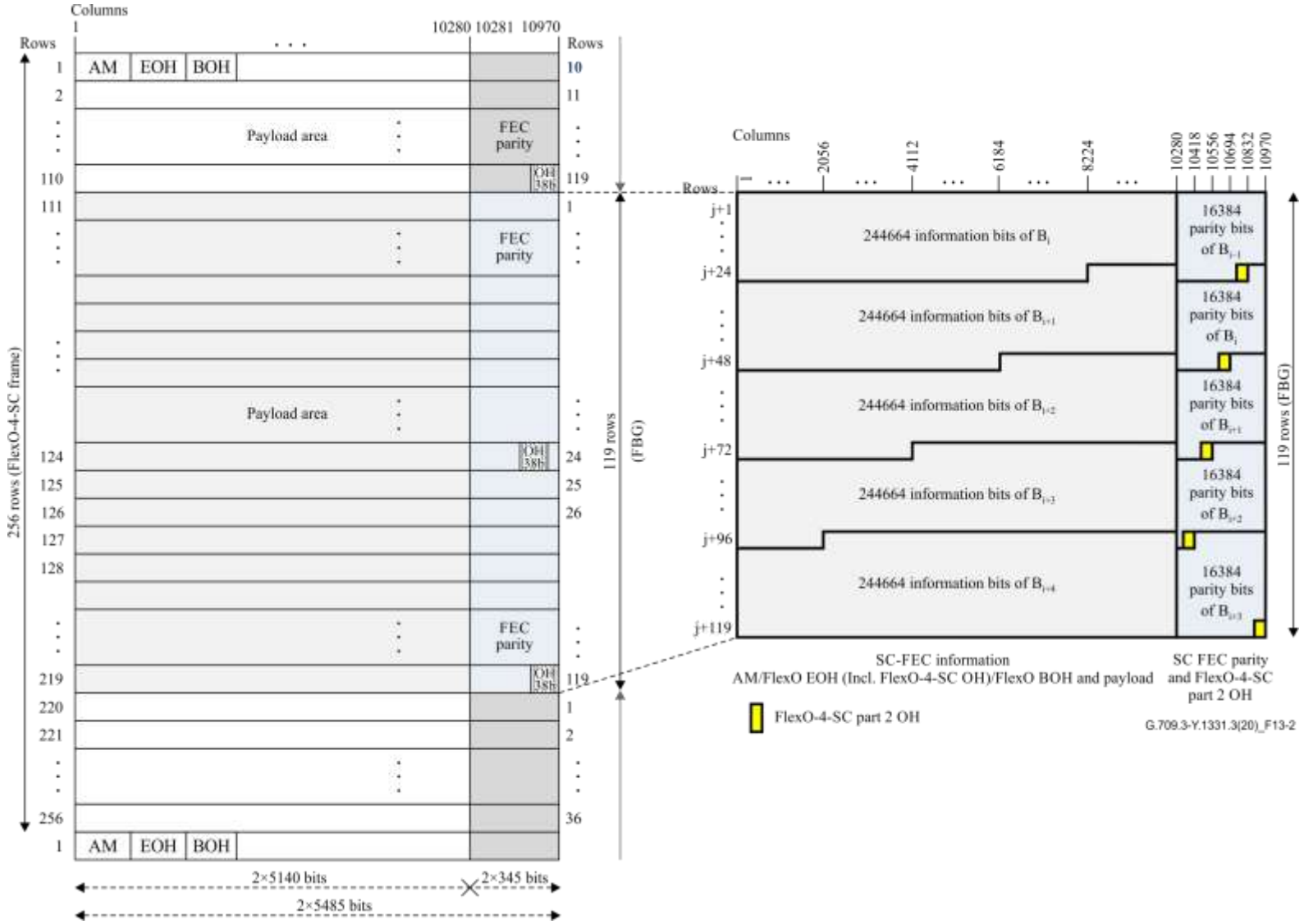


Figure 13-2 – FlexO-4-SCs FBG structure (left) and information and parity block and FlexO-4-SC OH boundaries (right)

13.2 FlexO-4-SC bit rate and frame periods

The bit rate and tolerance of the FlexO-4-SC signal is defined in Table 13-1.

Table 13-1 – FlexO-4-SC types and bit rates

FlexO-4-SC nominal bit rate	Bit-rate tolerance
524366/462961 × 4 × 99 532 800 kbit/s	±20 ppm
NOTE 1 – The nominal FlexO-4-SC bit rate is approximately: 450 937 475.984 kbit/s. NOTE 2 – The FlexO-4-SC bit rate can be based on the OTUC bit rate as follows: 4388/4097 × 4 × OTUC bit rate = 4388/4097 × 4 × 239/226 × 99 532 800 kbit/s. NOTE 3 – The FlexO-4-SC bit rate can be based on the 400G FlexO-4-RS bit rate as follows: 1097/1088 × 400G FlexO-4-RS bit rate = 1097/1088 × 256/241 × 4 × 239/226 × 99 532 800 kbit/s.	

The frame and multi-frame periods of the FlexO-4-SC signal are defined in Table 13-2.

Table 13-2 – FlexO-4-SC frame and multi-frame periods

Frame period (Note)	Multi-frame period (Note)
~6.228 µs	49.822 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.	

13.3 FlexO-4-SC overhead

The FlexO-4-SC frame contains two overhead areas. The first overhead area is located in the first 8 bits of the FlexO EOH in row 1, columns 1921 to 1928. It includes information to support the FBG alignment function. The second overhead area is located in the FlexO-4-SC FEC parity area, in blocks of 38 bits each between two successive SC FEC blocks.

The FlexO-4-SC OH is terminated where the FlexO-4-SC frame is assembled and disassembled.

An overview of the first part of the FlexO-4-SC OH area is presented in Figure 13-3. An overview of the second part of the FlexO FEC OH area is presented in Figure 13-2 and Figure 11-4.

13.3.1 FlexO-4-SC FEC block alignment (FBA)

Reserved for future use. Not used in FlexO-4-DSH interfaces.

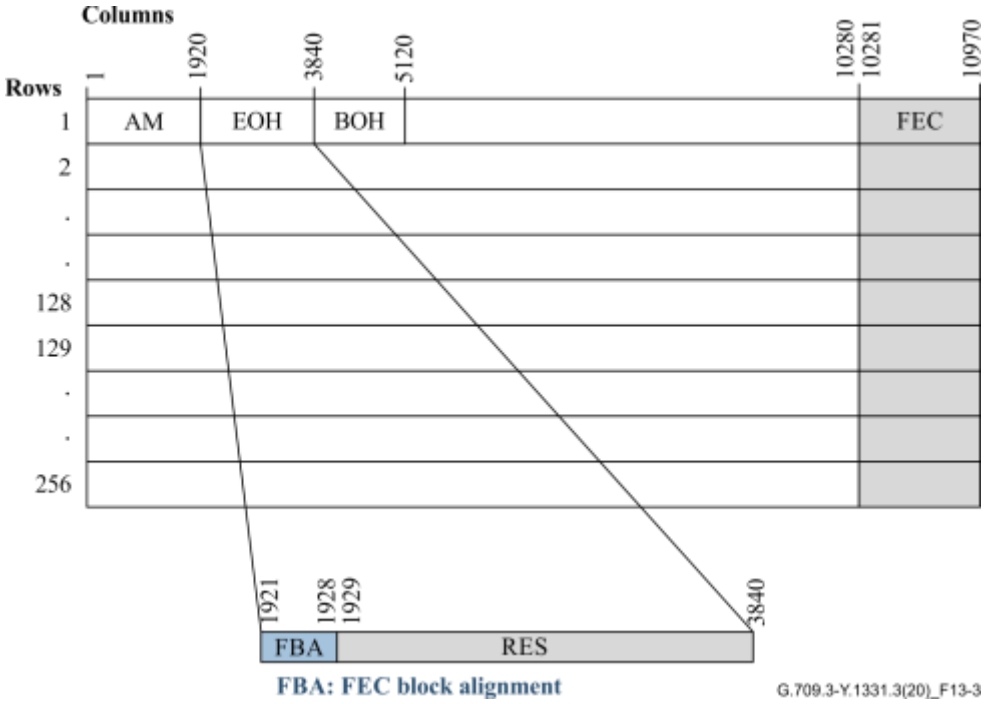


Figure 13-3 – FlexO-4-SC overhead, part 1

13.3.2 Multi block alignment signal (MBAS)

Refer to clause 11.3.2.

13.4 Staircase forward error correction (SC FEC)

The FlexO-4-SC FEC code is a 512-bit × 510-bit generalized staircase code (SC FEC) that works in conjunction with an error de-correlator. The error de-correlator function is used to randomize the error locations, in order to reduce impact of correlated errors on the decoder performance of the random error correcting SC FEC. The SC FEC code is systematic, and the 6.7% FlexO-4-SC FEC area specified in clause 13.1 is used to store the parity information generated by the encoder.

The generic operation of the staircase FEC scheme (with error de-correlator) is specified in Annex A. The FlexO-4-SC specific aspects of the staircase FEC operation are specified in Annex C.

14 FlexO-x-D<fec>

FlexO-x-D<fec> has a basic frame structure and a client and FEC specific multi-frame structure.

14.1 FlexO-x-D<fec> frame and multi-frame structures

The FlexO-x-D<fec> frame structure is shown in Figure 14-1 and consists of 116 rows by 32 Z-bit columns with Z = 8, 4. It contains a training sequence (TS) field in row 1, columns 1 to 11, a pilot sequence (PS) field in column 1 of rows 2 to 116 and a (21 + 115 × 31 = 3586 Z-bit) payload and forward error correction parity field in the remainder of the frame. The first Z-bit TS overhead block in row 1 column 1 is also used as 116th Z-bit PS block.

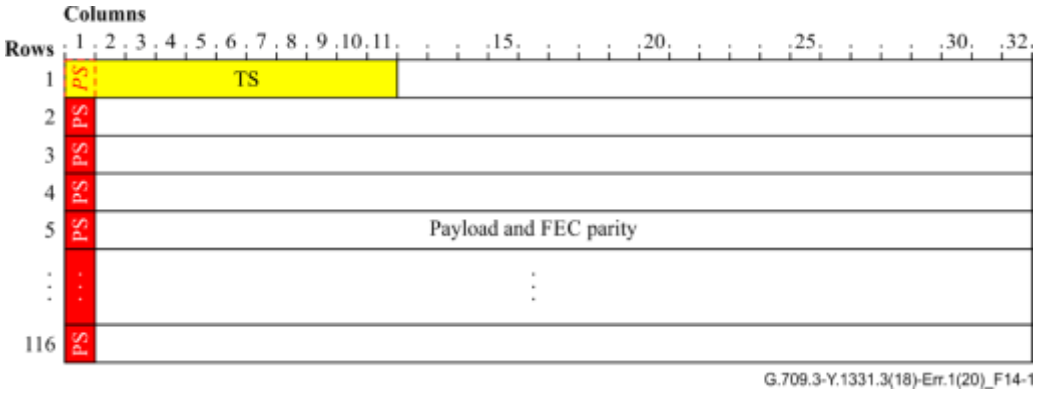


Figure 14-1 – FlexO-x-D<fec> frame structure

This frame structure can also be presented in a rowcolumn format as shown in Figure 14-2. The 3712 rowcolumns in this presentation represent the (116 × 32) Z-bit blocks of one frame. The rowcolumn number is determined by the following equation:

$$\text{rowcolumn\#} = (32 \times (\text{row\#} - 1)) + \text{column\#}$$

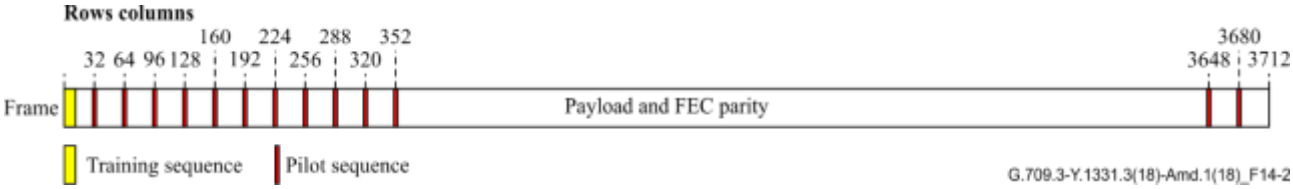


Figure 14-2 – FlexO-x-D<fec> 49-frame multi-frame structure alternative representation

This frame structure can be deployed in an N-frame multi-frame structure, of which the value of N is client and FEC specific. Refer to Figure 14-3 for an illustration of such multi-frame structure.

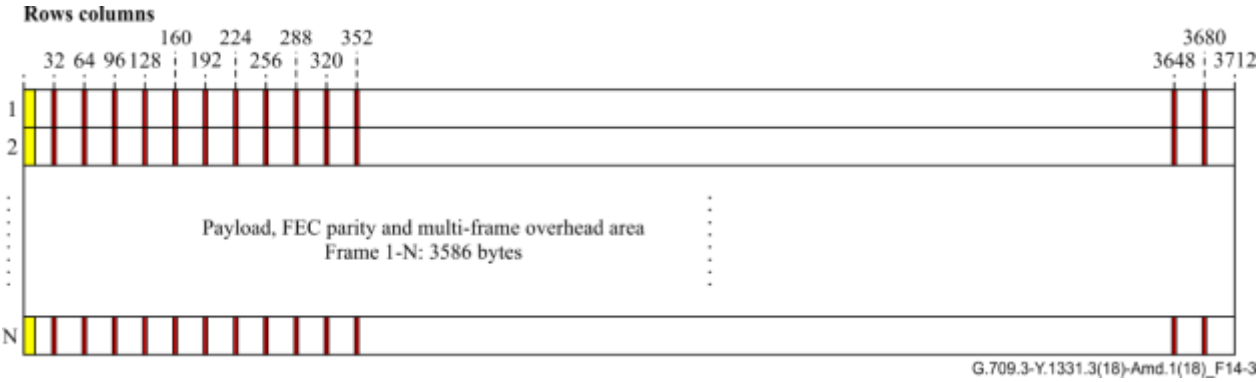


Figure 14-3 – FlexO-x-D<fec> N-frame multi-frame structure

14.2 FlexO-x-D<fec> Overhead

The FlexO-x-D<fec> frame has the following overhead included: training sequence and pilot sequence.

14.2.1 Training sequence (TS)

Training sequence (TS) overhead is used for FlexO-x-D<fec> frame alignment. The TS overhead consists of 11 Z-bit blocks as illustrated in Figure 14-4. The first Z-bit block has a value such that it can also be used as a 116th Z-bit PS overhead block.

The values of the 11 TS overhead blocks are specified in Annex G.

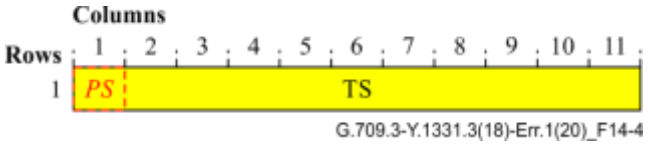


Figure 14-4 – Training sequence overhead

14.2.2 Pilot sequence (PS)

Pilot sequence (PS) overhead is used for FlexO-x-D<fec> frame alignment. The PS overhead consists of 1+115 Z-bit blocks as illustrated in Figure 14-5. The first Z-bit block has a value such that it can also be used as a 1st TS overhead block.

The values of the 1+115 PS overhead blocks are specified in Annex G.

Figure 14-5 – Pilot sequence overhead

14.2.2.1 8-bit block (Z=8)

The values of the 116 8-bit PS overhead blocks are specified in Annex G.

14.2.2.2 4-bit block (Z=4)

The values of the 116 4-bit PS overhead blocks are specified in Annex G.

15 FlexO-x-DSH

The FlexO-x-DSH signal format deploys the FlexO-x-D<fec> frame and carries a FlexO-x-SC client with an additional Hamming SD FEC.

15.1 FlexO-x-DSH multi-frame and super-frame structures

15.1.1 Multi-frame structure

The FlexO-x-DSH multi-frame structure is shown in Figure 15-1 and consists of 49 FlexO-x-D<fec> frames. It includes an additional 22 Z-bit multi-frame alignment signal (MFAS) located in frame 1, row 1 columns 12 to 32, and row 2 column 2 to identify the 49 frame multi-frame and a 76 Z-bit fixed stuff field (FS) in frame 1, row 2 columns 3 to 32, row 3 columns 2 to 32 and row 4 columns 2 to 16.

The multi-frame payload area contains 3488 + 48 × 3586 = 175616 Z-bit blocks. The number of bits and 128-bit blocks supported by these Z-bit blocks are presented in Table 15-1.

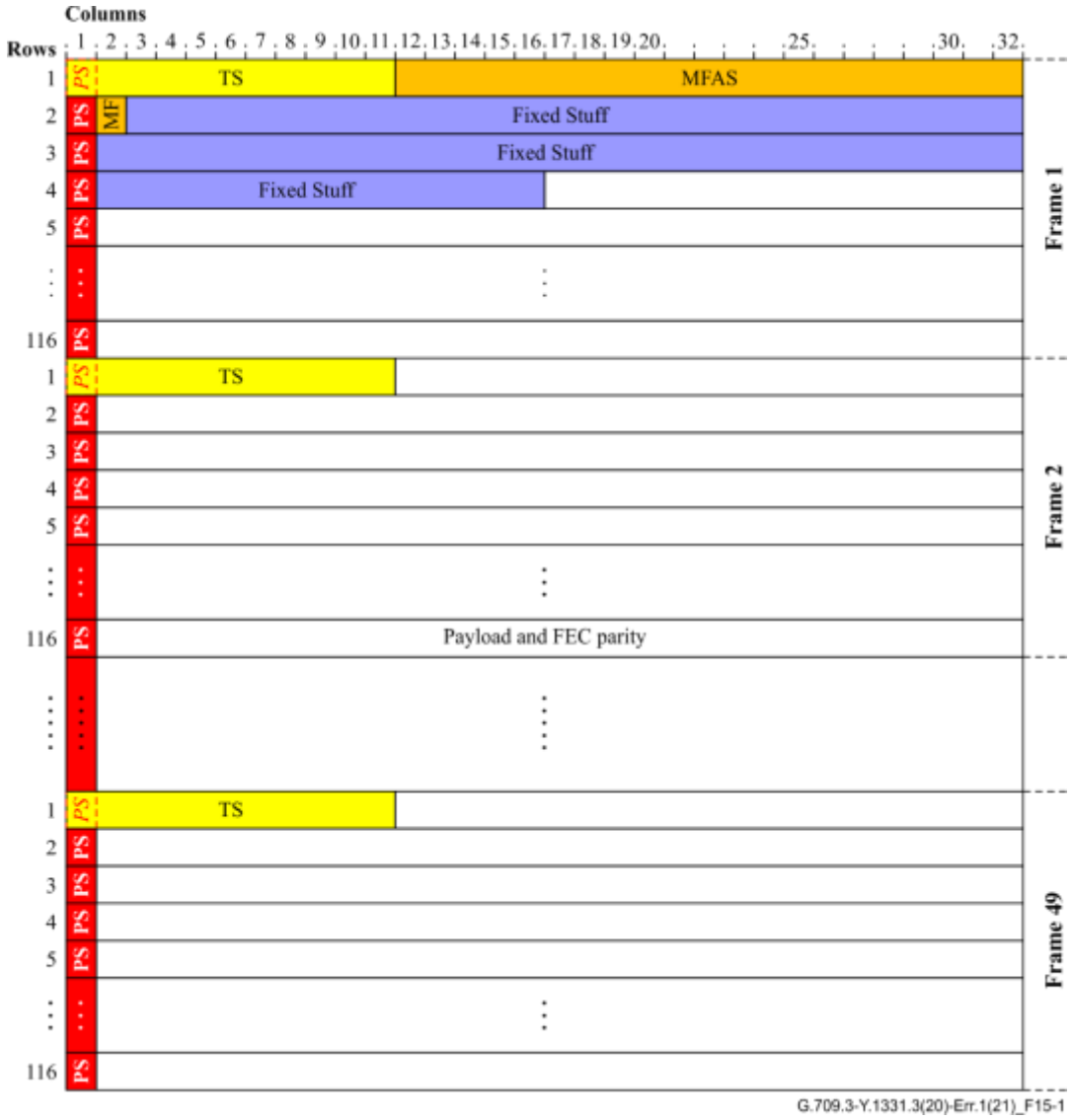


Figure 15-1 – FlexO-x-DSH 49-frame multi-frame structure

This multi-frame structure can also be presented in a frame and rowcolumn format as shown in Figure 15-2. The 3712 rowcolumns in this presentation represent the (116 × 32) Z-bit blocks of one frame. The rowcolumn number is determined by the following equation:

$$\text{rowcolumn\#} = (32 \times (\text{row\#} - 1)) + \text{column\#}.$$

Every frame contains 11 Z-bit blocks of training sequence overhead and 115 Z-bit blocks of pilot overhead. The first frame contains furthermore 22 Z-bit MFAS blocks and 76 Z-bit FS blocks. The first frame contains 3488 Z-bit payload and FEC parity blocks, the other frames contain 3586 Z-bit payload and parity blocks.

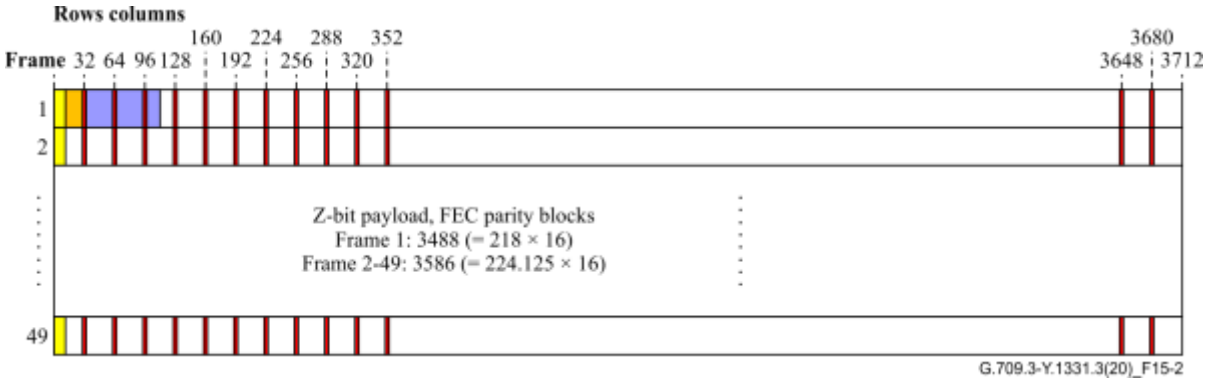


Figure 15-2 – FlexO-x-DSH 49-frame multi-frame structure alternative representation

15.1.2 Super-frame structure

The FlexO-x-DSH super-frame structure is shown in Figure 15-3 and consists of four multi-frames. It includes an additional 22 Z-bit super-frame alignment signal (SFAS) located in the first 22 Z-bit blocks of the multi-frame's FS overhead field (in frame 1, row 2 columns 3 to 24).

Figures 15-3 illustrates the 4-multi-frame super-frame structure. Each multi-frame contains an SFAS overhead field to hold a unique SFAS sequence with the values ABBB as specified in clause 15.3.2.

NOTE – It is possible to perform super-frame alignment only. Achieving super-frame alignment implicitly results in multi-frame alignment. For such case, the super-frame behaves like a 196-frame multi-frame.

For Z = 8, one multi-frame is able to carry one FBG[2] and one super-frame carries four FBGs[1].

For Z = 4, two multi-frames are able to carry one FBG[1] and one super-frame carries two FBGs[1].

[2] Refer to clause 15.4.5 for the implication of the convolutional interleaver process.

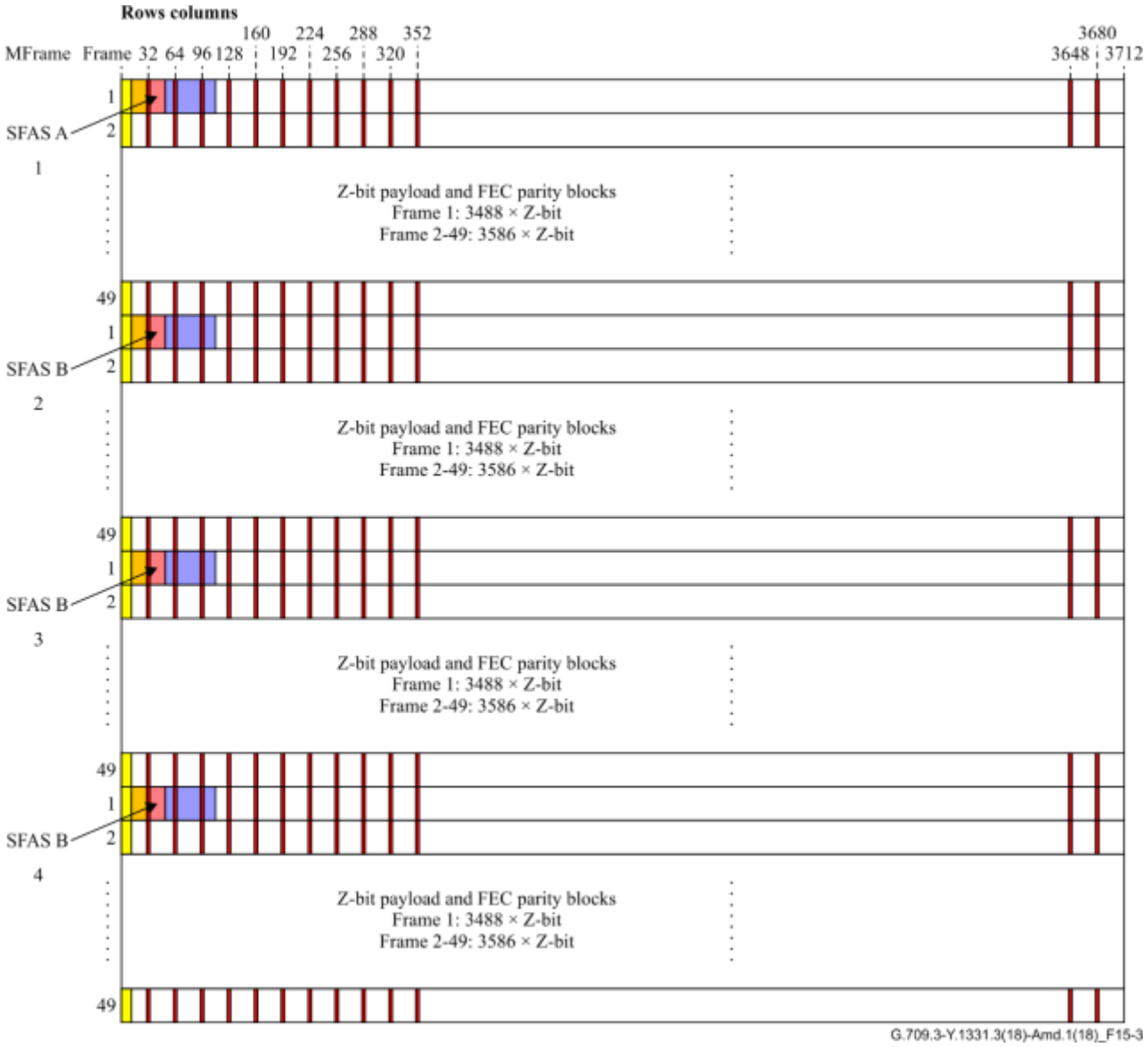


Figure 15-3 – FlexO-x-DSH 4-multi-frame super-frame structure

15.1.3 Payload and FEC parity area

The FlexO-x-DSH multi-frame payload and FEC parity field contains 10976 × Z/8 × 16-byte (128-bit) blocks of which the first 119 bits carry a 119-bit block of the FlexO-x-SC client and the last 9 bits carry a 9-bit Hamming soft decision (HSD) FEC parity as illustrated in Figure 15-4.

The 4-multi-frame super-frame contains 10976 × Z/8 × 4 of such 128-bit blocks.

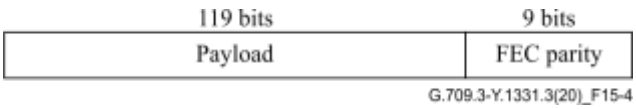


Figure 15-4 – 128-bit payload and FEC parity block

The FlexO-x-DSH multi- and super-frame payload and FEC parity areas do not divide elegantly into 128-bit blocks in a single frame. The 128-bit block will spill over and cross frame boundaries, while it will not cross multi-frame boundaries for x is 2 to 4.

Table 15-1 – Number of bits and 128-bit blocks supported by 175616 Z-bit blocks in a multi-frame

Z	Number of multi-frame payload and FEC parity bits	Number of 16-byte/128-bit payload and FEC parity bit blocks per multi-frame	Number of multi-frames to carry one FBG (Note)	Number of FBGs in 4-multi-frame super-frame (Note)
8	1,404,928	10976	1	4
4	702,464	5488	2	2
NOTE – Refer to 15.4.5 for FBG mapping details and implication of the convolutional interleaver process.				

15.2 FlexO-x-DSH bit rates and frame periods

The bit rates and tolerance of the FlexO-x-DSH signals are defined in Table 15-2.

The bit rate and tolerance of the FlexO-x-DSH payload and FEC parity area signals are defined in Table 15-3.

The frame and multi-frame periods of the FlexO-x-DSH signals are defined in Table 15-4.

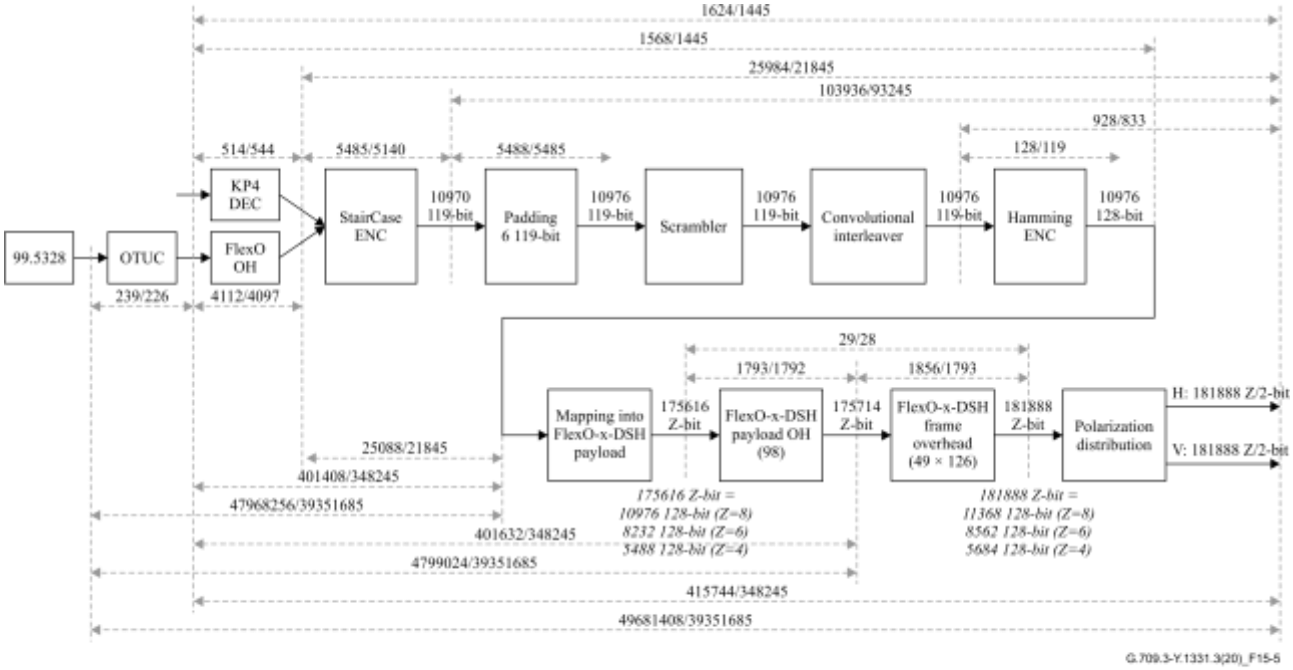


Figure 15-5 – FlexO-x-DSH processes and bit rate ratios

Table 15-2 – FlexO-x-DSH types and bit rates

FlexO-x-DSH type	FlexO-x-DSH nominal bit rate	Bit-rate tolerance
100G FlexO-1-DSH	49681408/39351685 × 1 × 99 532 800 kbit/s	±20 ppm
200G FlexO-2-DSH	49681408/39351685 × 2 × 99 532 800 kbit/s	
400G FlexO-4-DSH	49681408/39351685 × 4 × 99 532 800 kbit/s	
NOTE 1 – The nominal FlexO-x-DSH bit rates are approximately: 125 659 921.454 kbit/s (FlexO-1-DSH), 251 319 842.908 kbit/s (FlexO-2-DSH) and 502 639 685.816 kbit/s (FlexO-4-DSH). NOTE 2 – The FlexO-x-DSH bit rates can be based on the OTUC bit rate as follows: 415744/348245 × x × OTUC bit rate. NOTE 3 – The FlexO-x-DSH bit rates can be based on the FlexO-x bit rates as follows: 25984/21845 × FlexO-x bit rate. NOTE 4 – The FlexO-x-DSH bit rates can be based on the FlexO-x-RS bit rates as follows: 1624/1445 × FlexO-x-RS bit rate.		

Table 15-3 – FlexO-x-DSH types and payload and FEC parity area bit rates

FlexO-x-DSH type	FlexO-x-DSH nominal payload and FEC area bit rate	Bit-rate tolerance
100G FlexO-1-DSH	47968256/39351685 × 1 × 99 532 800 kbit/s	±20 ppm
200G FlexO-2-DSH	47968256/39351685 × 2 × 99 532 800 kbit/s	
400G FlexO-4-DSH	47968256/39351685 × 4 × 99 532 800 kbit/s	
NOTE 1 – The nominal FlexO-x-DSH payload and FEC area bit rates are approximately: 121 326 820.714 kbit/s (FlexO-1-DSH), 242 653 641.428 kbit/s (FlexO-2-DSH) and 485 307 282.857 kbit/s (FlexO-4-DSH). NOTE 2 – The FlexO-x-DSH payload and FEC area bit rates can be based on the OTUC bit rate as follows: 401408/348245 × x × OTUC bit rate. NOTE 3 – The FlexO-x-DSH payload and FEC area bit rates can be based on the FlexO-x bit rates as follows: 25088/21845 × FlexO-x bit rate. NOTE 4 – The FlexO-x-DSH payload and FEC area bit rates can be based on the FlexO-x-RS bit rates as follows: 1568/1445 × FlexO-x-RS bit rate.		

Table 15-4 – FlexO-x-DSH frame and multi-frame periods

Bit rate	Frame period (Note)	Multi-frame period (Note)	Super-frame period (Note)
100G FlexO-1-DSH (Z=4)	~0.118 µs	~5.790 µs	~23.160 µs
200G FlexO-2-DSH (Z=8)	~0.118 µs	~5.790 µs	~23.160 µs
200G FlexO-2-DSH (Z=4)	~0.059 µs	~2.895 µs	~11.580 µs
400G FlexO-4-DSH (Z=8)	~0.059 µs	~2.895 µs	~11.580 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.			

15.3 Overhead

The FlexO-x-DSH 49-frame multi-frame has the following overhead included: multi-frame alignment signal and fixed stuff.

The FlexO-x-DSH 4-multi-frame super-frame has the following overhead included: super-frame alignment signal.

15.3.1 Multi-frame alignment signal (MFAS)

Multi-frame alignment signal (MFAS) overhead is used for FlexO-x-DSH 49-frame multi-frame alignment. The MFAS overhead consists of 22 Z-bit blocks as illustrated in Figure 15-6.

NOTE 1 – This same MFAS overhead is used also for the FlexO-x-DO 48-frame multi-frame alignment; refer to clause 16.3.1.

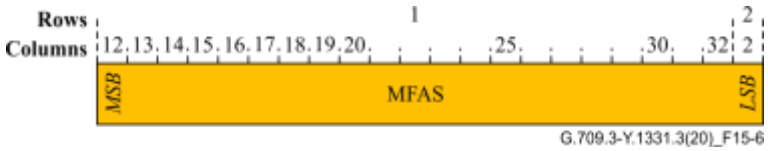


Figure 15-6 – N-frame (N=48,49) multi-frame alignment signal overhead

The values of the 22 MFAS overhead Z-bit blocks are specified in Annex G.

15.3.2 Super-frame alignment signal (SFAS)

Super-frame alignment signal (SFAS) overhead is used for FlexO-x-DSH 4-multi-frame super-frame alignment. The SFAS overhead consists of 22 Z-bit blocks as illustrated in Figure 15-7.

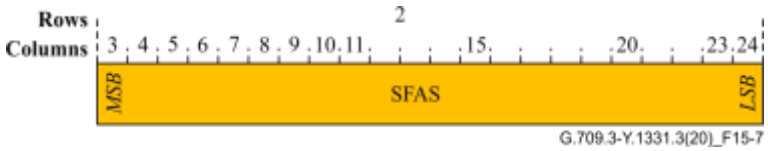


Figure 15-7 – 4-multi-frame super-frame alignment signal overhead

The values of the 22 SFAS overhead bytes in multi-frame #1 and #2..4 are specified in Table 15-6.

Table 15-6 – SFAS A and B sequence values

	SFAS values (Z=8)		SFAS values (Z=4)	
	A (MF #1)	B (MF #2..4)	A (MF #1)	B (MF #2..4)
(Row,Column) #	1234 5678	1234 5678	12 34	12 34
(2,3)	1000 0110	0001 0000	00 01	10 01
(2,4)	1101 1110	1010 1111	10 00	10 00
(2,5)	1011 1100	0110 1100	10 11	00 00
(2,6)	1000 1101	1100 0100	00 11	01 00
(2,7)	1111 1110	0100 0111	01 01	10 00
(2,8)	0001 1110	0000 1111	01 00	11 00
(2,9)	1100 1101	1011 0110	10 00	01 11
(2,10)	0010 1011	1110 1101	01 00	11 01
(2,11)	0100 0010	0011 1101	10 01	01 10
(2,12)	0000 1100	0111 0010	01 00	10 10
(2,13)	1010 0000	1001 0000	11 11	01 00
(2,14)	1001 1000	1001 1100	11 11	10 01
(2,15)	0011 0001	1110 1010	00 10	00 10
(2,16)	0101 0111	1000 1000	01 10	11 11
(2,17)	0111 0000	1111 1001	11 10	01 00
(2,18)	0010 0110	0001 0011	11 01	11 10
(2,19)	0011 1110	1110 0100	00 11	01 11

Table 15-6 – SFAS A and B sequence values

(Row,Column) #	SFAS values (Z=8)		SFAS values (Z=4)	
	A (MF #1)	B (MF #2..4)	A (MF #1)	B (MF #2..4)
	1234 5678	1234 5678	12 34	12 34
(2,20)	0100 1011	1010 0110	10 01	11 10
(2,21)	1111 0101	0100 0010	11 00	01 01
(2,22)	1111 1110	0101 0001	00 10	01 10
(2,23)	1000 0001	1011 0110	01 11	11 10
(2,24)	0101 1110	1101 0111	10 10	11 00

15.3.3 Fixed stuff (FS)

54 Z-bit blocks are specified as FS per multi-frame as illustrated in Figure 15-8. These Z-bit blocks should contain a randomized bit pattern.

Figure 15-8 – Fixed stuff overhead

15.4 Mapping of FlexO-x-SC client into FlexO-x-DSH payload

The FlexO-x-SC signal is the client signal of FlexO-x-DSH signal and the bits of one FlexO-x-SC signal are carried in the payload area of one FlexO-x-DSH signal.

The bits of an FBG plus six 119-bit PAD blocks, after scrambling, interleaving (in blocks of 119 bits) by a convolutional interleaving process and (128,119) Hamming soft decision FEC encoding are mapped into the payload and FEC parity area of a FlexO-x-DSH signal.

The convolutional interleaving process serves to spread out the transmission order of consecutive blocks of 119 bits from the staircase FEC encoded frame, to increase the resilience of the bit stream to error bursts. Consequentially, the 119-bit blocks of the FlexO-x-SC signal within the FlexO-x-DSH multi- or super-frame payload do not appear in sequential order and the 119-bit blocks within one FlexO-x-DSH multi- or super-frame payload are from two (or more) successive FBGs.

15.4.1 Scrambling

The FlexO-x-SC signal extended with 714 PAD bits every FBG is scrambled. The operation of the scrambler shall be functionally equivalent to that of a frame-synchronous scrambler of sequence 65535 and the generating polynomial shall be $x^{16} + x^{12} + x^{3} + x + 1$. See Figure 15-9.

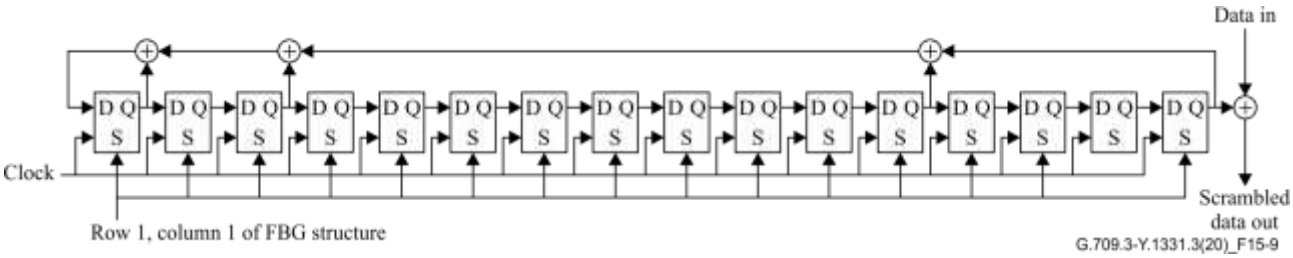


Figure 15-9 – Frame synchronous scrambler

The scrambler resets to 0xFFFF on row 1, column 1 of the FBG structure as shown in Figures 15-10, 15-11, and 15-12. The scrambler state advances during each bit of the FBG structure and subsequent 714-bit PAD block.

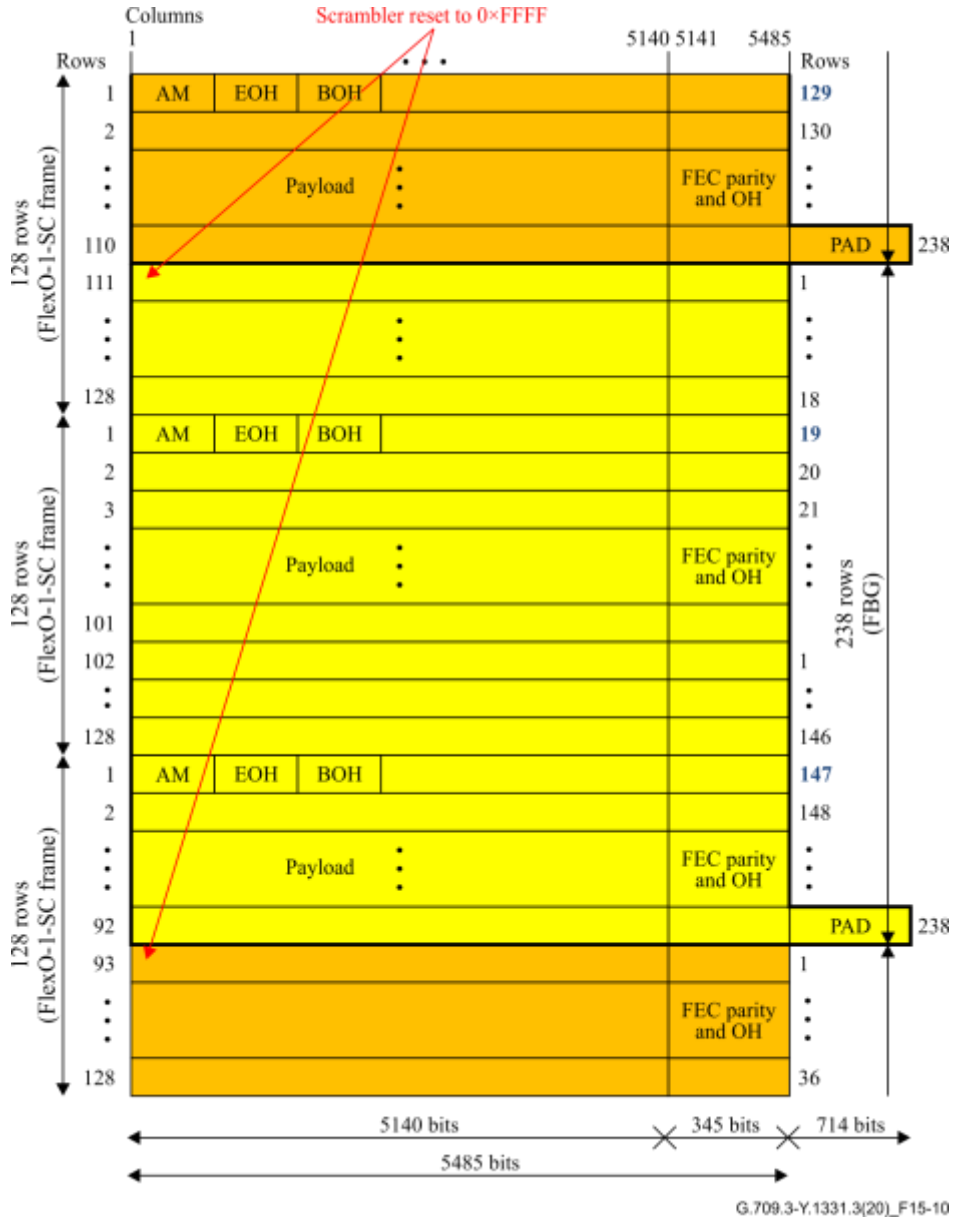


Figure 15-10 – FlexO-1-SC bit stream scrambling with frame synchronous scrambler

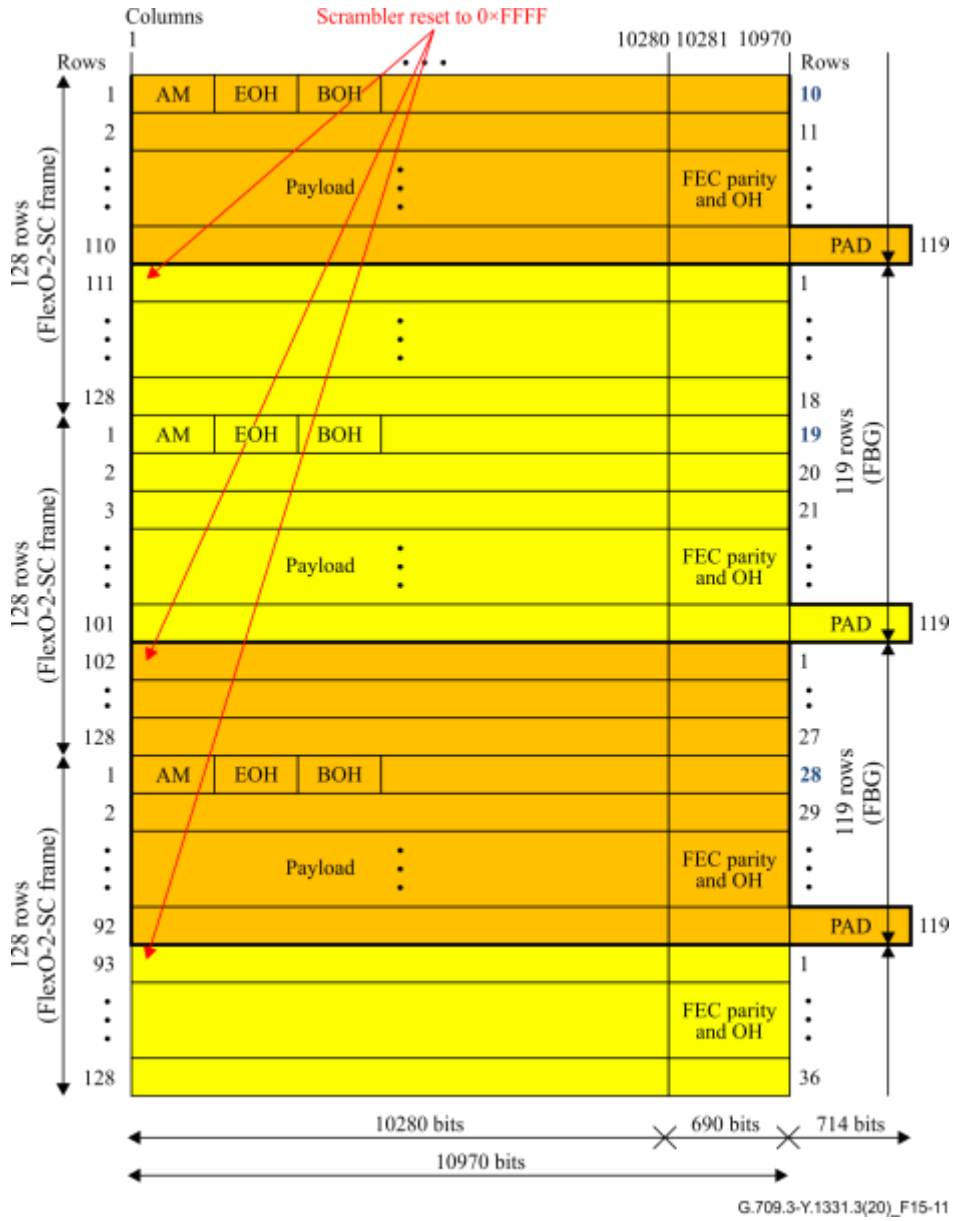


Figure 15-11 – FlexO-2-SC bit stream scrambling with frame synchronous scrambler

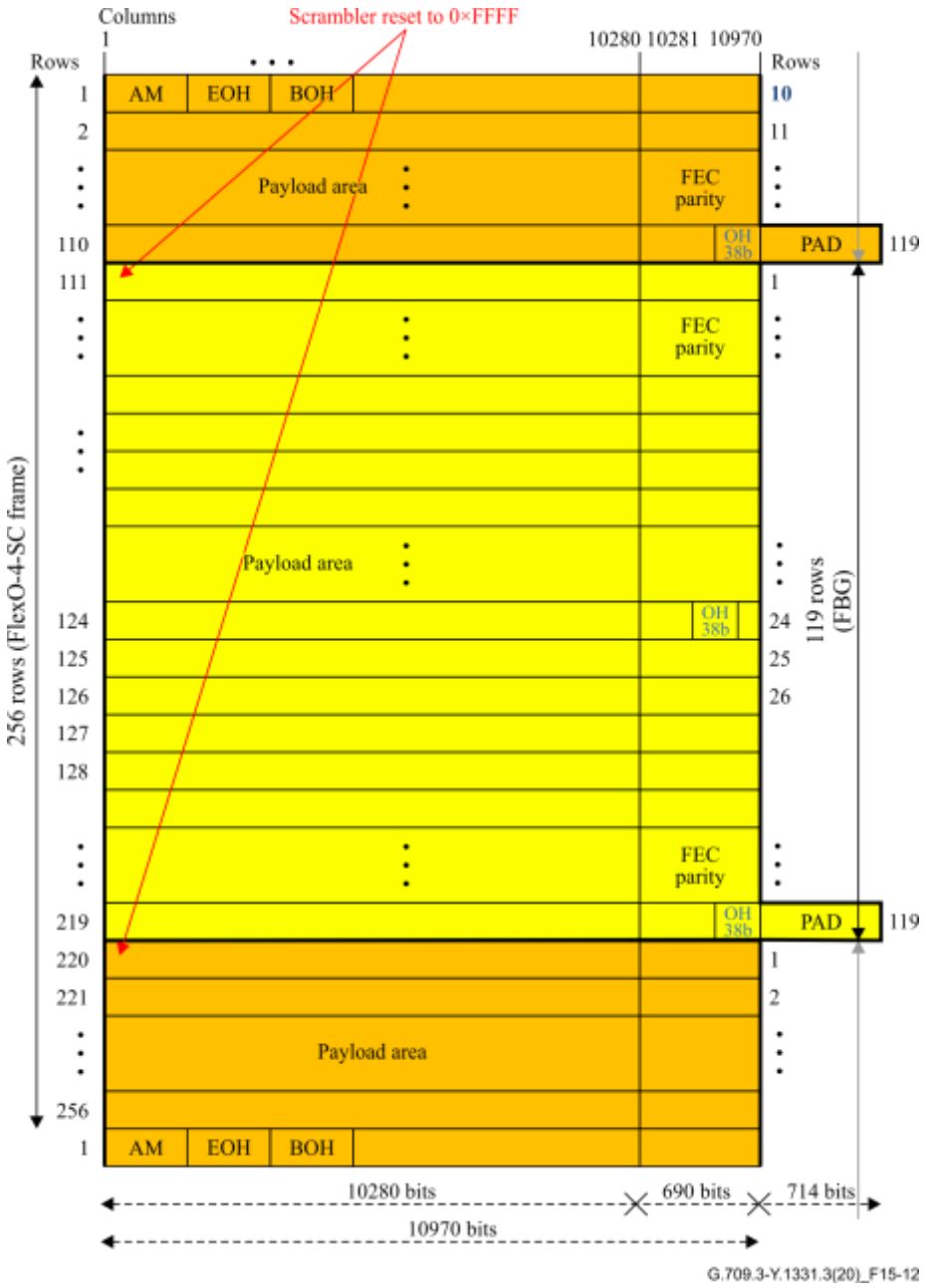


Figure 15-12 – FlexO-4-SC bit stream scrambling with frame synchronous scrambler

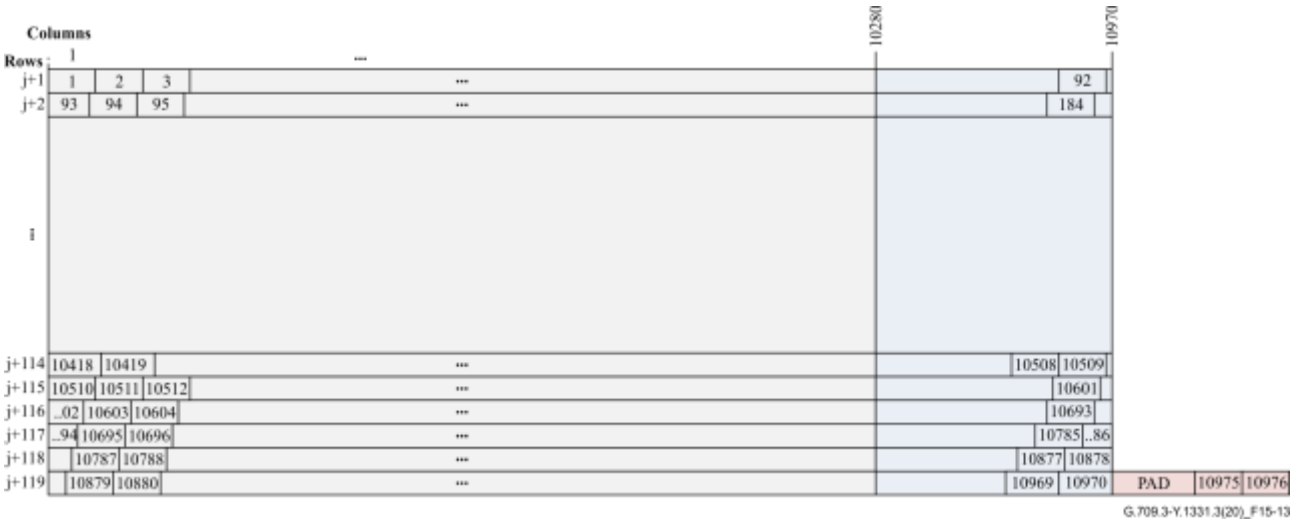


Figure 15-13 – 119-bit block numbering within FlexO-1-SC, FlexO-2-SC and FlexO-4-SC FBG plus 714-bit PAD block

15.4.2 119-bit FlexO-x-SC and PAD block numbering

The 119-bit blocks within a FlexO-x-SC frame are numbered from 1 to 10970 and the six additional PAD blocks are numbered 10971 to 10976 as shown in Figure 15-13.

119-bit block #1 contains the first 119 bits of the FBG structure within a FlexO-x-SC signal; these are the bits in row j+1, columns 1 to 119. 119-bit block #2 contains the second 119 bits of the FBG structure, which are the bits in row j+1, columns 120 to 238,etc. 119-bit block 10970 contains the last 119 bits of the FBG structure, which are the bits in row j+119, columns 10852 to 10970.

15.4.3 Convolutional interleaving

The convolutional interleaving process is of depth 16, and consists of 16 parallel delay lines (numbered 0 to 15), as illustrated in Figure 15-11. Each delay operator "D" represents a storage element of 119-bit (119b). From one delay line to the next lower delay line, two delays operators are deleted.

1) At time i, the input and output switches are aligned at row b_i:
2) A block of 119b is read from row b_i
3) The contents of row b_i are shifted to the right by 119b
4) A block of 119b is written to row b_i

The switch position is updated to $b_{i+1} = (b_i + 1)\ mod\ 16$.

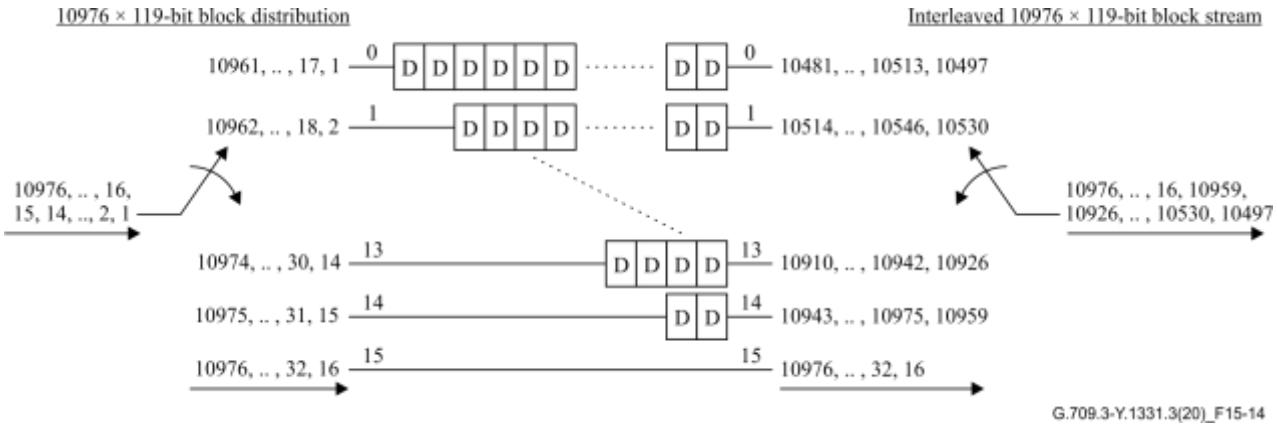


Figure 15-14 – Convolutional interleaving of 10976 119-bit blocks

Initialization of the convolutional interleaving switches (to their topmost positions) is defined to occur at the start of every FBG plus 714 bit PAD block structure. Since this structure has 10976 119-bit blocks and 10976 is evenly divisible by the depth of the convolutional interleaving process (i.e., 16), the switches will wraparound to this position at the start of every FBG plus 714 bit PAD block structure, as illustrated in Figure 15-16. 119-bit block #1 is applied at delay line #0, 119-bit block #2 at delay line #2, etc. up to 119-bit block #10976 at delay line #15.

The convolutional interleaving process presents the 10976 119-bit blocks in an interleaved order at its output ports. When 119-bit blocks #1 to #16 are presented at the 16 input ports, the 16 output ports present 119-bit blocks #10497, #10530, .., #10926, #10959 and #16, as illustrated in Figure 15-15 and Table-15-7.

Note that blocks #10497, # 10530, .., #10926 and #10959 are from the previous FBG plus 714 PAD block structure, while block #16 is from the current FBG plus 714 PAD block structure.

Table 15-8 illustrates the effect of the convolutional interleaving process. Each row in this table represents 16 119-bit blocks presented at the 16 input ports.

– The last row represents the time at which 119-bit blocks #1 to #16 are presented at the 16 input ports and the diagonal blue line in the middle represents the 119-bit blocks that are presented at the 16 output ports at that time.

– The last but one row represents the time at which 119-bit blocks #10961 to #10976 are presented at the 16 input ports and the diagonal black line in the middle represents the 119-bit blocks that are presented at the 16 output ports at that time.

– Block # 1 is the ($30 \times 16 + 1 =$) 481^{st} block in the sequence of 10976 blocks.

Table 15-7 – 119-bit blocks carried in two successive FlexO-x-DSH multi-frames

10481	10482	10483	10484	10485	10486	10487	10488	10489	10490	10491	10492	10493	10494	10495	10496
10497	10498	10499	10500	10501	10502	10503	10504	10505	10506	10507	10508	10509	10510	10511	10512
10513	10514	10515	10516	10517	10518	10519	10520	10521	10522	10523	10524	10525	10526	10527	10528
10529	10530	10531	10532	10533	10534	10535	10536	10537	10538	10539	10540	10541	10542	10543	10544
10545	10546	10547	10548	10549	10550	10551	10552	10553	10554	10555	10556	10557	10558	10559	10560
10561	10562	10563	10564	10565	10566	10567	10568	10569	10570	10571	10572	10573	10574	10575	10576
10577	10578	10579	10580	10581	10582	10583	10584	10585	10586	10587	10588	10589	10590	10591	10592
10593	10594	10595	10596	10597	10598	10599	10600	10601	10602	10603	10604	10605	10606	10607	10608
10609	10610	10611	10612	10613	10614	10615	10616	10617	10618	10619	10620	10621	10622	10623	10624
10625	10626	10627	10628	10629	10630	10631	10632	10633	10634	10635	10636	10637	10638	10639	10640
10641	10642	10643	10644	10645	10646	10647	10648	10649	10650	10651	10652	10653	10654	10655	10656
10657	10658	10659	10660	10661	10662	10663	10664	10665	10666	10667	10668	10669	10670	10671	10672
10673	10674	10675	10676	10677	10678	10679	10680	10681	10682	10683	10684	10685	10686	10687	10688
10689	10690	10691	10692	10693	10694	10695	10696	10697	10698	10699	10700	10701	10702	10703	10704
10705	10706	10707	10708	10709	10710	10711	10712	10713	10714	10715	10716	10717	10718	10719	10720
10721	10722	10723	10724	10725	10726	10727	10728	10729	10730	10731	10732	10733	10734	10735	10736
10737	10738	10739	10740	10741	10742	10743	10744	10745	10746	10747	10748	10749	10750	10751	10752
10753	10754	10755	10756	10757	10758	10759	10760	10761	10762	10763	10764	10765	10766	10767	10768
10769	10770	10771	10772	10773	10774	10775	10776	10777	10778	10779	10780	10781	10782	10783	10784
10785	10786	10787	10788	10789	10790	10791	10792	10793	10794	10795	10796	10797	10798	10799	10800
10801	10802	10803	10804	10805	10806	10807	10808	10809	10810	10811	10812	10813	10814	10815	10816
10817	10818	10819	10820	10821	10822	10823	10824	10825	10826	10827	10828	10829	10830	10831	10832
10833	10834	10835	10836	10837	10838	10839	10840	10841	10842	10843	10844	10845	10846	10847	10848
10849	10850	10851	10852	10853	10854	10855	10856	10857	10858	10859	10860	10861	10862	10863	10864
10865	10866	10867	10868	10869	10870	10871	10872	10873	10874	10875	10876	10877	10878	10879	10880
10881	10882	10883	10884	10885	10886	10887	10888	10889	10890	10891	10892	10893	10894	10895	10896
10897	10898	10899	10900	10901	10902	10903	10904	10905	10906	10907	10908	10909	10910	10911	10912
10913	10914	10915	10916	10917	10918	10919	10920	10921	10922	10923	10924	10925	10926	10927	10928
10929	10930	10931	10932	10933	10934	10935	10936	10937	10938	10939	10940	10941	10942	10943	10944
10945	10946	10947	10948	10949	10950	10951	10952	10953	10954	10955	10956	10957	10958	10959	10960
10961	10962	10963	10964	10965	10966	10967	10968	10969	10970	*10971*	*10972*	*10973*	*10974*	*10975*	*10976*
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

The convolutional de-interleaving process presents the 10976 119-bit blocks in a de-interleaved order at its output ports. When 119-bit blocks #10497, #10530, .., #10926, #10959 and #16 are presented at the 16 input ports, the 16 output ports present 119-bit blocks #10497 to #10512, as illustrated in Figure 15-15.

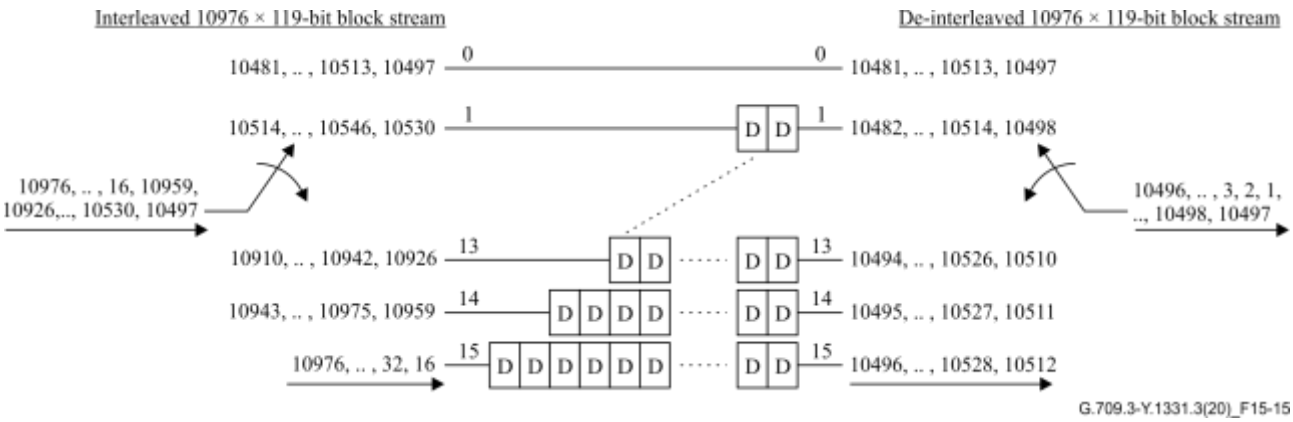


Figure 15-15 – Convolutional de-interleaving of 10976 119-bit blocks

15.4.4 Hamming soft decision forward error correction (DSH FEC)

The FlexO-x-DSH FEC is a systematic (128,119) double-extended Hamming soft decision code. The generic operation of the Hamming soft decision FEC scheme is specified in Annex D. It adds 9-bits of parity to each of the 10976 119-bit blocks as output by the convolutional interleaving process and results in 10976 128-bit blocks.

15.4.5 Mapping

15.4.5.1 Mapping into FlexO-x-DSH for Z=8 (DP-16QAM symbols)

Figure 15-16 illustrates the mapping of the 128-bit blocks within the FlexO-x-DSH (Z=8) frame, multi-frame and super-frame prior to symbol interleaving.

Frame 1, row 4, columns 17 to 32 contain 128-bit block #10497. Frame 1, row 5, columns 2 to 17 contain block #10530. Etc. Frame 49, row 116, columns 17 to 32 contain 128-bit block #10976.

Block #1 is the 481st 128-bit block and is located in frame 3, row 20, columns 29 to 32 and row 21, columns 2 to 13. There is as such an offset of (2×3712 + 19×32 + 28 =) 8060 bytes between the start of FlexO-4-DSH multi-frame and this 128-bit block #1.

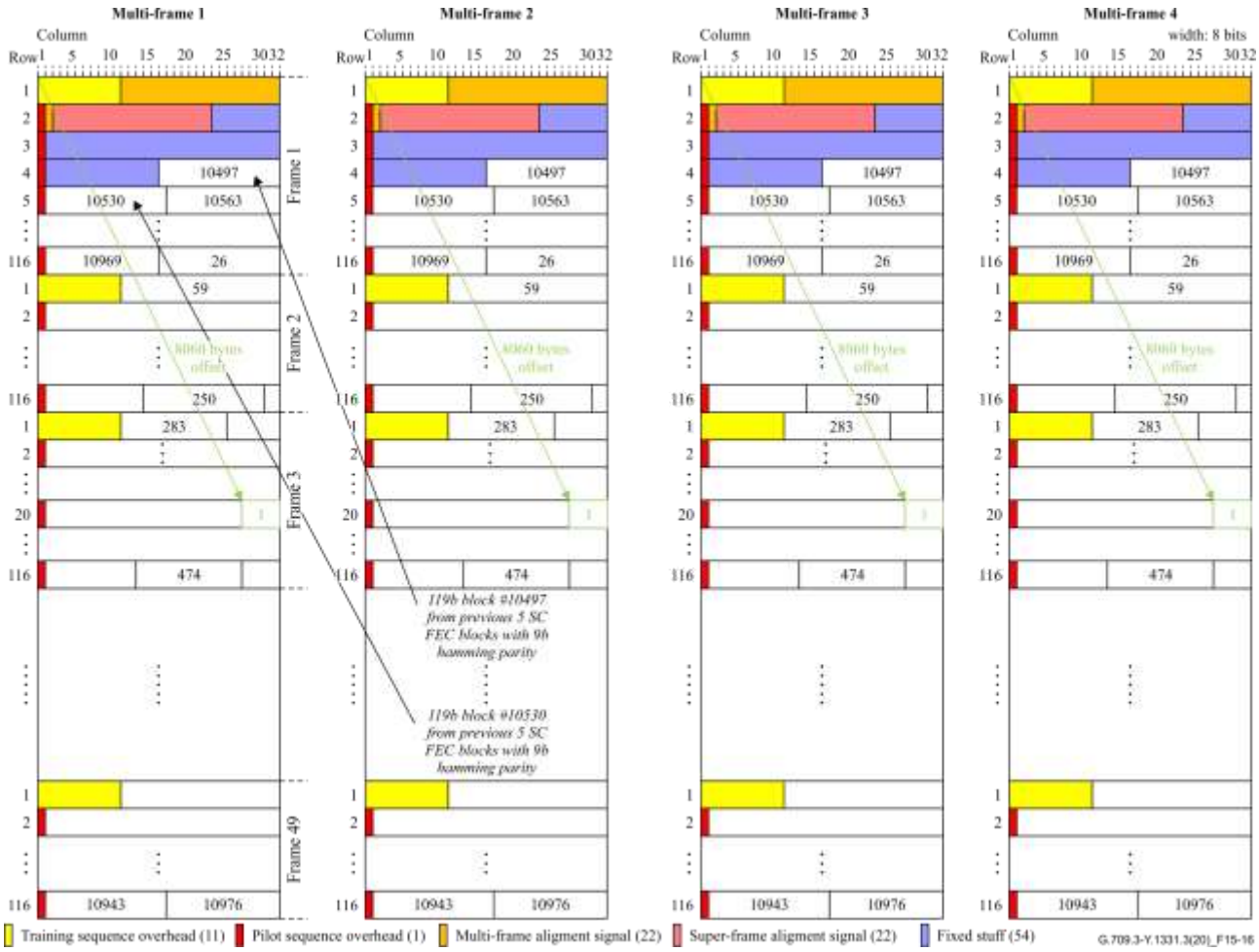


Figure 15-16 – Mapping of the 4 × 10976 128-bit blocks into FlexO-x-DSH for Z=8 super-frame

DP-16QAM Symbol (Z=8) mapping and polarization distribution

Each 128-bit code word (S_j, $j \geq 0$) is mapped to 16 DP-16QAM symbols ($S_{j,i}$, $j \geq 0$, i = 0..15),

$$S_j = [S_{j,0}, S_{j,1}, \ldots, S_{j,15}]$$

where,

– (c_{8i},c_{8i+1}) maps to the in-phase (I) component of the X-pol of $S_{j,i}$

– (c_{8i+2},c_{8i+3}) maps to the quadrature-phase (Q) component of the X-pol of $S_{j,i}$

– (c_{8i+4},c_{8i+5}) maps to the I component of the Y-pol of $S_{j,i}$

– (c_{8i+6},c_{8i+7}) maps to the Q component of the Y-pol of $S_{j,i}$

Interleaving DP-16QAM Symbols

The DP-16QAM symbols are time-interleaved to de-correlate the noise between consecutively received symbols, as well as to uniformly distribute the symbols (mapped from a single Hamming code word) between pilot symbols.

Prior to MFAS and PS insertion, each frame consists of 10976×16 DP-16QAM symbols. The symbol interleave interleaves the 16 symbols $S_{j,0}$, $S_{j,1}$, $S_{j,2}$, .., $S_{j,15}$ from 8 consecutive Hamming code words S_j, S_{j+1}, S_{j+2}, .., S_{j+7} into the sequence $S_{j,0}$, $S_{j+1,0}$, $S_{j+2,0}$, .., $S_{j+7,0}$, $S_{j,1}$, $S_{j+1,1}$, .., $S_{j+7,1}$, $S_{j,2}$, .., $S_{j+7,15}$.

This represents byte interleaving of 8 successive 128-bit Hamming code words in the mapping.

15.4.5.2 Mapping into FlexO-x-DSH for Z=4 (DP-QPSK symbols)

Figure 15-17 illustrates the mapping of the 128-bit blocks within the FlexO-x-DSH (Z=4) frame, multi-frame and super-frame prior to symbol interleaving.

Frame 1, row 4, columns 17 to 32 and row 5, columns 2 to 17 contain 128-bit block #10497. Frame 1, row 5, columns 18 to 32 and row 6, columns 2 to 18 contain block #10530. Etc.

Block #1 is the 481st 128-bit block and is located in frame 5, row 37, columns 10 to 32 and row 37, columns 2 to 10. There is as such an offset of (4×3712 + 36×32 + 9 = 16009 Z-bit (Z=4) blocks =) 8004.5 bytes between the start of FlexO-x-DSH (Z=4) frame and this 128-bit block #1.

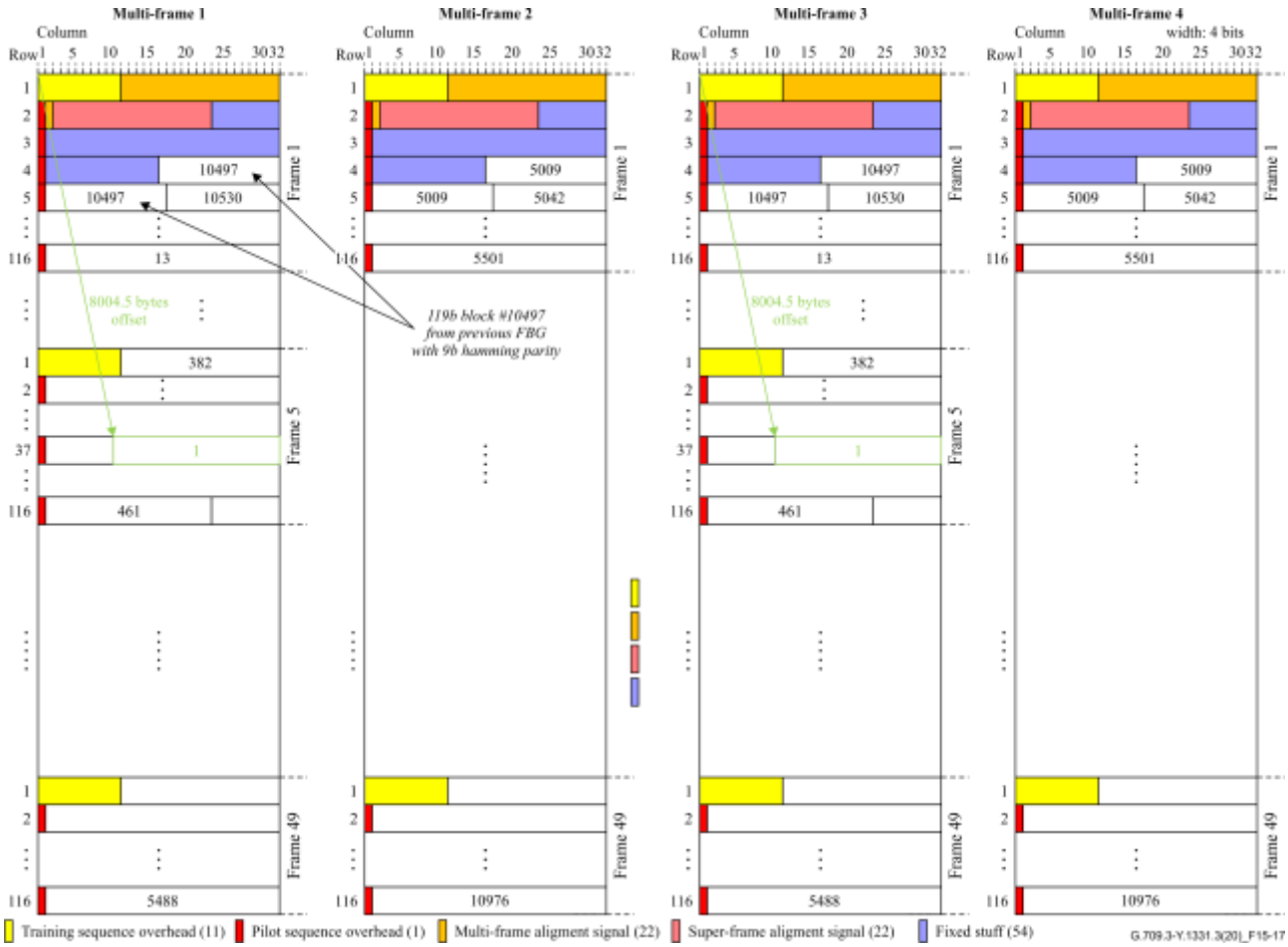


Figure 15-17 – Mapping of the 2 × 10976 128-bit blocks into FlexO-x-DSH for Z=4 super-frame

DP-QPSK Symbol mapping and polarization distribution

Each 128-bit code word (S_j, $j \geq 0$) is mapped to 32 DP-QPSK symbols ($S_{j,i}$, $j \geq 0$, i = 0..31),

$S_j=[S_{j,0},S_{j,1},\ldots,S_{j,31}]$

where,

– (c_{8i}) maps to the in-phase (I) component of the X-pol of $S_{j,i}$

– (c_{8i+1}) maps to the quadrature-phase (Q) component of the X-pol of $S_{j,i}$

– (c_{8i+2}) maps to the I component of the Y-pol of $S_{j,i}$

– (c_{8i+3}) maps to the Q component of the Y-pol of $S_{j,i}$

Interleaving DP-QPSK Symbols

The DP-QPSK symbols are time-interleaved, to de-correlate the noise between consecutively received symbols, as well as to uniformly distribute the symbols (mapped from a single Hamming code word) between pilot symbols.

Prior to fame alignment word (FAW) and pilot insertion, each 2-frame multi-frame consists of 10976×32 DP-QPSK symbols. The symbol interleave interleaves the 32 symbols $S_{j,0}$, $S_{j,1}$, $S_{j,2}$, .., $S_{j,31}$ from 4 consecutive Hamming code words S_j, S_{j+1}, S_{j+2}, S_{j+3} into the sequence $S_{j,0}$, $S_{j+1,0}$, $S_{j+2,0}$, $S_{j+3,0}$, $S_{j,1}$, $S_{j+1,1}$, $S_{j+2,1}$, $S_{j+3,1}$, $S_{j,2}$, .., $S_{j+3,31}$.

This presents 4-bit nibble interleaving of 4 successive 128-bit Hamming code words in the mapping.

15.5 FOICx.k-DSH

A conceptually serial FlexO-x-DSH signal is adapted to a parallel multi-lane distribution (MLD) signal format with k lanes, referred to as FOICx.k-DSH.

15.5.1 FOIC2.4-DSH lanes

The FlexO-2-DSH (Z=4) bits are distributed to four logical FOIC2.4-DSH lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 4-bit block is carried on lane #i (i = 1..4). Each FOIC2.4-DSH lane is synchronous to the FlexO-2-DSH (Z=4) frame.

Each FlexO-2-DSH (Z=4) super-frame contains 4 × 3712 × 49 × 4 = 2910208 bits. Each FOIC2.4-DSH lane will carry 25% of these bits, which are 4 × 3712 × 49 = 727552 bits.

The bit rates and tolerance of the FOIC2.4-DSH lanes are defined in Table 15-8.

Table 15-8 – FOIC2.4-DSH types and bit rates

FOIC2.4-DSH lane nominal bit rate	Bit-rate tolerance
49681408/39351685 × 49 766 400 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC2.4-DSH lane bit rates is approximately: 62 829 960.727 kbit/s.	

15.5.2 FOIC2.8-DSH lanes

The FlexO-2-DSH (Z=8) bits are distributed to eight logical FOIC2.8-DSH lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 8-bit block is carried on lane #i (i = 1..8). Each FOIC2.8-DSH lane is synchronous to the FlexO-2-DSH (Z=8) frame.

Each FlexO-2-DSH (Z=8) super-frame contains 4 × 3712 × 49 × 8 = 5820416 bits. Each FOIC2.8-DSH lane will carry 12.5% of these bits, which are 4 × 3712 × 49 = 727552 bits.

The bit rates and tolerance of the FOIC4.8-DSH lanes are defined in Table 15-9.

Table 15-9 – FOIC2.8-DSH types and bit rates

FOIC2.8-DSH lane nominal bit rate	Bit-rate tolerance
49681408/39351685 × 24 883 200 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC4.8-DSH lane bit rate is approximately: 31 414 980.363 kbit/s.	

15.5.3 FOIC4.8-DSH lanes

The FlexO-4-DSH (Z=8) bits are distributed to eight logical FOIC4.8-DSH lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 8-bit block is carried on lane #i (i = 1..8). Each FOIC4.8-DSH lane is synchronous to the FlexO-4-DSH (Z=8) frame.

Each FlexO-4-DSH (Z=8) super-frame contains 4 × 3712 × 49 × 8 = 5820416 bits. Each FOIC4.8-DSH lane will carry 12.5% of these bits, which are 4 × 3712 × 49 = 727552 bits.

The bit rates and tolerance of the FOIC4.8-DSH lanes are defined in Table 15-10.

Table 15-10 – FOIC4.8-DSH types and bit rates

FOIC4.8-DSH lane nominal bit rate	Bit-rate tolerance
49681408/39351685 × 49 766 400 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC4.8-DSH lane bit rate is approximately: 62 829 960.727 kbit/s.	

15.5.4 FOIC1.4-DSH lanes

The FlexO-1-DSH (Z=4) bits are distributed to four logical FOIC1.4-DSH lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 4-bit block is carried on lane #i (i = 1..4). Each FOIC1.4-DSH lane is synchronous to the FlexO-1-DSH (Z=4) frame.

Each FlexO-1-DSH (Z=4) super-frame contains 4 × 3712 × 49 × 4 = 2910208 bits. Each FOIC1.4-DSH lane will carry 25% of these bits, which are 4 × 3712 × 49 = 727552 bits.

The bit rates and tolerance of the FOIC1.4-DSH lanes are defined in Table 15-11.

Table 15-11 – FOIC1.4-DSH types and bit rates

FOIC2.4-DSH lane nominal bit rate	Bit-rate tolerance
49681408/39351685 × 24 883200 kbit/s	±20 ppm
NOTE – The nominal FOIC1.4-DSH lane bit rates is approximately: 31 414 980.363 kbit/s.	

16 FlexO-x-DO

The FlexO-x-DO signal format deploys the FlexO-x-D<fec> frame and carries a FlexO-x client with a spatially coupled product-like forward error correction code, referred to as Open FEC (OFEC).

16.1 FlexO-x-DO multi-frame and super-frame structures

16.1.1 Multi-frame structure

The FlexO-x-DO multi-frame structure is shown in Figure 16-1 and consists of 48 FlexO-x-D<fec> frames, containing 48 × 3712 = 178176 Z-bit blocks.

It includes an additional 22 Z-bit multi-frame alignment signal (MFAS) located in frame 1, row 1 columns 12 to 32 and row 2 column 2 to identify the 48 frame multi-frame and a 74 Z-bit Fixed stuff field (FS) in frame 1, row 2 columns 3 to 32, row 3 columns 2 to 32 and row 4 columns 2 to 14.

The multi-frame payload and FEC parity area contains 3490 + 47 × 3586 = 172032 Z-bit blocks. The number of bits supported by these Z-bit blocks are presented in Table 16-1.

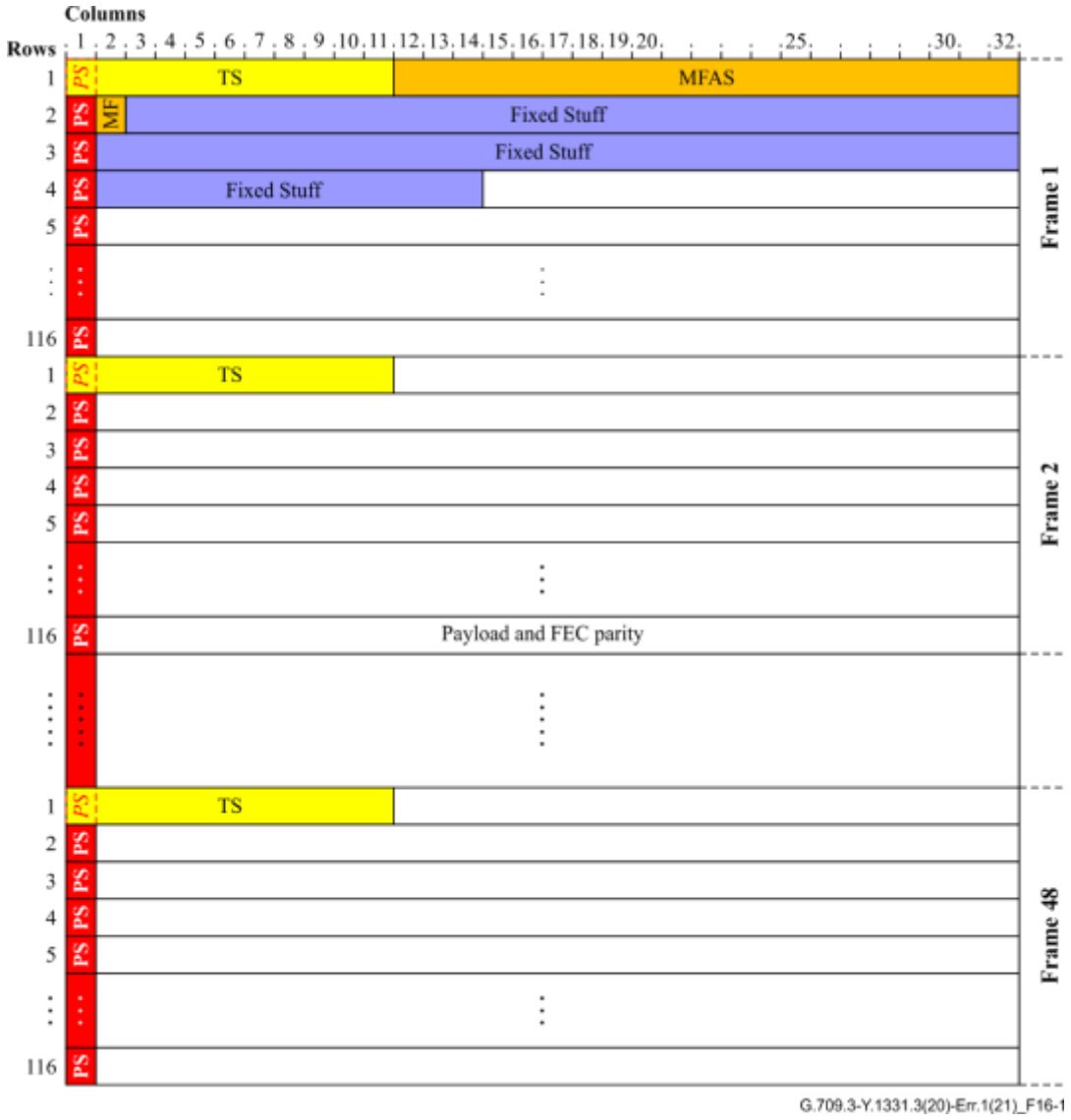


Figure 16-1 – FlexO-x-DO 48-frame multi-frame structure

This multi-frame structure can also be presented in a frame and rowcolumn format as shown in Figure 16-2. The 3712 rowcolumns in this presentation represent the (116 × 32) Z-bit blocks of one frame. The rowcolumn number is determined by the following equation:

$$\text{rowcolumn\#} = (32 \times (\text{row\#} - 1)) + \text{column\#}.$$

Every frame contains 11 Z-bit blocks of training sequence overhead and 115 Z-bit blocks of pilot overhead. The first frame contains furthermore 22 Z-bit MFAS blocks and 74 Z-bit Fixed stuff blocks. The first frame contains 3490 Z-bit payload and FEC parity blocks, the other frames contain 3586 Z-bit payload and parity blocks.

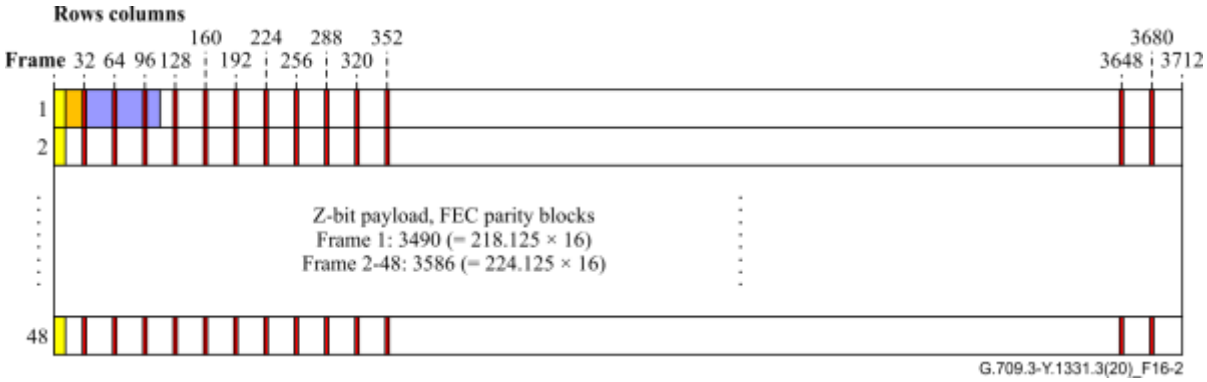


Figure 16-2 – FlexO-x-DO 48-frame multi-frame structure alternative representation

16.1.2 Payload and FEC parity area

The FlexO-x-DO multi-frame payload and FEC parity field contains 172032 × Z-bit blocks. One multi-frame is able to carry one OFEC Block Group (OFBG) that contains 149184 Z-bit blocks. 111 OFBG bits are complemented with 17 FEC parity bits. OFBG and parity bits are distributed throughout the FlexO-x-DO multi-frame payload and FEC parity area as specified in clause 16.4.

Table 16-1 – Number of bits supported by 172032 Z-bit blocks in a multi-frame

Z	Number of payload and FEC parity bits	Number of payload bits	Number of 111|128|256|257-bit payload blocks	Number of FEC parity bits
8	1376256	1193472	10752 | 9324 | 4662 | 4640	182784
4	688128	596736	5376 | 4662 | 2331 | 2320	91392

16.2 FlexO-x-DO bit rates and frame periods

The bit rates and tolerance of the FlexO-x-DO signals are defined in Table 16-2.

The bit rate and tolerance of the FlexO-x-DO payload and FEC parity area signals are defined in Table 16-3.

The frame and multi-frame periods of the FlexO-x-DO signals are defined in Table 16-4.

The bit rate ratio associated with the various FlexO-x-DO processes are illustrated in Figure 16-3.

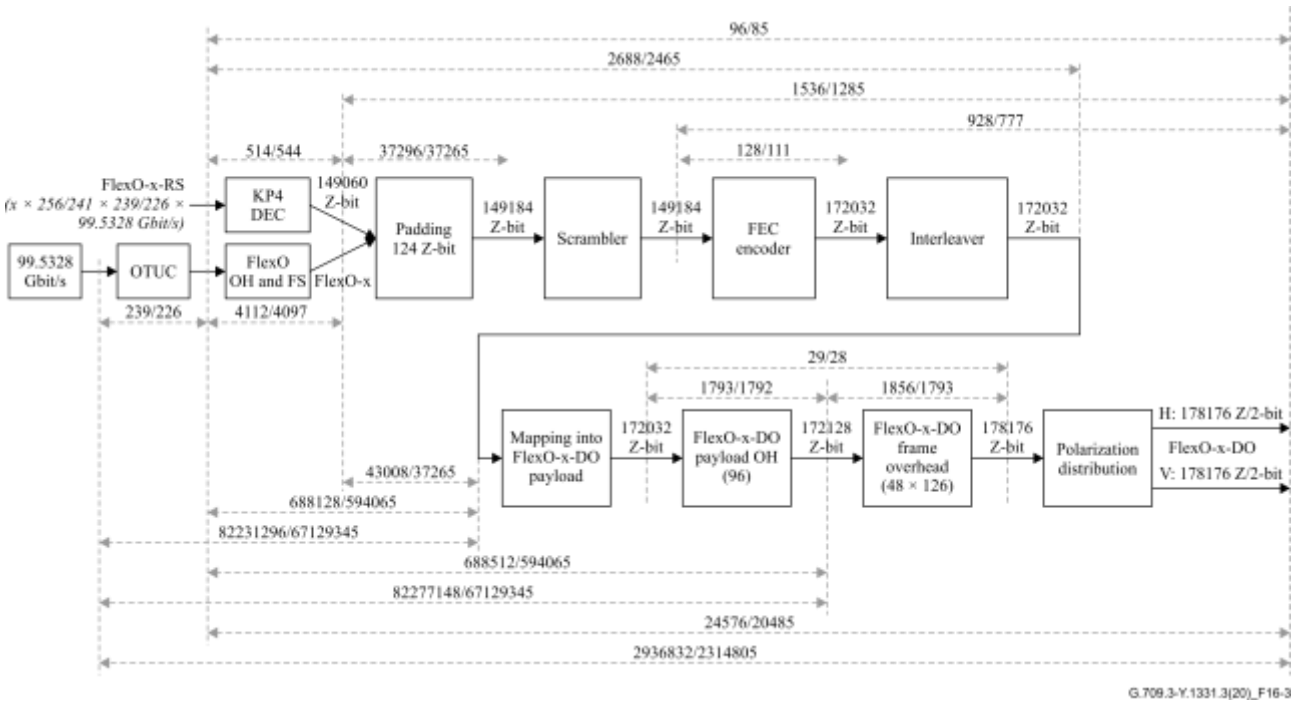


Figure 16-3 – FlexO-x-DO processes and bit rate ratios

Table 16-2 – FlexO-x-DO types and bit rates

FlexO-x-DO type	FlexO-x-DO nominal bit rate	Bit-rate tolerance
100G FlexO-1-DO	2936832/2314805 × 1 × 99 532 800 kbit/s	±20 ppm
200G FlexO-2-DO	2936832/2314805 × 2 × 99 532 800 kbit/s	
400G FlexO-4-DO	2936832/2314805 × 4 × 99 532 800 kbit/s	
NOTE 1 – The nominal FlexO-x-DO bit rates are approximately: 126 278 935.845 kbit/s (FlexO-1-DO), 252 557 871.691 kbit/s (FlexO-2-DO) and 505 115 743.382 kbit/s (FlexO-4-DO). NOTE 2 – The FlexO-x-DO bit rates can be based on the OTUC bit rate as follows: 24576/20485 × x × OTUC bit rate. NOTE 3 – The FlexO-x-DO bit rates can be based on the FlexO-x bit rates as follows: 1536/1285 × FlexO-x bit rate. NOTE 4 – The FlexO-x-DO bit rates can be based on the FlexO-x-RS bit rates as follows: 96/85 × FlexO-x-RS bit rate.		

Table 16-3 – FlexO-x-DO types and payload and FEC area bit rates

FlexO-x-DO type	FlexO-x-DO nominal payload and FEC area bit rate	Bit-rate tolerance
100G FlexO-1-DO	28/29 × FlexO-1-DO nominal bit rate	±20 ppm
200G FlexO-2-DO	28/29 × FlexO-2-DO nominal bit rate	
400G FlexO-4-DO	28/29 × FlexO-4-DO nominal bit rate	
NOTE 1 – The nominal FlexO-x-DO payload and FEC area bit rates are approximately: 121 924 489.782 kbit/s (FlexO-1-DO), 243 848 979.564 kbit/s (FlexO-2-DO) and 487 697 959.127 kbit/s (FlexO-4-DO). NOTE 2 – The FlexO-x-DO payload and FEC area bit rates can be based on the OTUC bit rate as follows: 688128/594065 × x × OTUC bit rate. NOTE 3 – The FlexO-x-DO payload and FEC area bit rates can be based on the FlexO-x bit rates as follows: 43008/37265 × FlexO-x bit rate. NOTE 4 – The FlexO-x-DO payload and FEC area bit rates can be based on the FlexO-x-RS bit rates as follows: 2688/2465 × FlexO-x-RS bit rate.		

Table 16-4 – FlexO-x-DO frame and multi-frame periods

Bit rate	Frame period (Note)	Multi-frame period (Note)
100G FlexO-1-DO (Z=4)	~0.118 µs	~5.644 µs
200G FlexO-2-DO (Z=8)	~0.118 µs	~5.644 µs
200G FlexO-2-DO (Z=4)	~0.059 µs	~2.822 µs
400G FlexO-4-DO (Z=8)	~0.059 µs	~2.822 µs
NOTE – The period is an approximated value, rounded to 3 decimal places.		

16.3 Overhead

The FlexO-x-DO 48-frame multi-frame has the following overhead included: multi-frame alignment signal and fixed stuff.

16.3.1 Multi-frame alignment signal (MFAS)

Multi-frame alignment signal (MFAS) overhead is used for FlexO-x-DO 48-frame multi-frame alignment. The MFAS overhead consists of 22 Z-bit blocks as illustrated in Figure 15-6.

The values of the 22 MFAS overhead Z-bit blocks are specified in Annex G.

16.3.2 Fixed stuff (FS)

74 Z-bit blocks are specified as fixed stuff per multi-frame as illustrated in Figure 16-4. These Z-bit blocks should contain a randomized bit pattern.

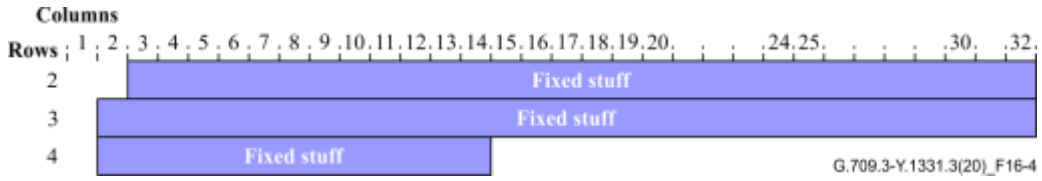


Figure 16-4 – Fixed stuff overhead

16.4 Mapping of FlexO-x client into FlexO-x-DO payload

The FlexO-x signal is the client signal of FlexO-x-DO signal and the bits of one FlexO-x signal are carried in the payload and FEC parity area of the FlexO-x-DO signal together with FEC parity bits.

The bits of an OFBG plus 124 Z-bit PAD blocks, after scrambling, FEC encoding and interleaving are mapped into the payload and FEC parity area of a FlexO-x-DO signal.

The interleaver process serves to spread out the transmission order of OFBG and FEC parity bits to increase the resilience of the bit stream to error bursts. The bit stream is interleaved to de-correlate the noise between consecutive symbols in the deinterleaving process and to uniformly distribute those symbols.

16.4.1 OFEC block group (OFBG)

To accommodate the block length of the OFEC, which is not aligned with the FlexO-x frame length, an additional OFEC block group (OFBG) structure is superimposed on the underlying FlexO-x multi-frame structure. The OFBG is aligned to the FlexO-x multi-frame rows and contains the bits of consecutive FlexO-x multi-frame rows plus some additional, appended pad bits, as illustrated in Figures 16-5 to 16-8.

The pad bits are all-zero and get scrambled prior to encoding, and removed after decoding and descrambling.

Table 16-5 specifies the number of consecutive FlexO-x multi-frame rows and pad bits that belong to one OFBG.

An OFBG is carried in the payload and FEC parity area of a FlexO-x-DO multi-frame, therefore the number of FlexO-x and pad bits of an OFBG is dependent on Z. This dependency is represented by OFBGz (z = 8,4). The number of FlexO-x rows in an OFBGz is dependent on "x".

An OFBGz is divided in a 21 × Z 7104-bit OFEC Coder (OFC) blocks, numbered 0 to 21×Z-1. The boundaries of these 21 × Z OFC blocks within an OFBG are illustrated in the right segment of Figures 16-5 to 16-8.

OFC block #0 occupies the 1st 7104 bits in the OFBG8 (row 1, columns 1 to 7104), OFC block #1 occupies the 2nd 7104 bits in the OFBG8 (row 1, columns 7105 to 10280, row 2, columns 1 to 3928), etc.

OFC block #0 occupies the 1st 7104 bits in the OFBG4 within a FlexO-2 (row 1, columns 1 to 7104), OFC block #1 occupies the 2nd 7104 bits in the OFBG4 (row 1, columns 7105 to 10280, row 2, columns 1 to 3928), etc.

OFC block #0 occupies the 1st 7104 bits in the OFBG4 within a FlexO-1 (row 1 and row 2, columns 1 to 1964), OFC block #1 occupies the 2nd 7104 bits in the OFBG4 (row 2, columns 1965 to 5140 and row 3, columns 1 to 3928), etc.

Table 16-5 – OFEC Block Group superimposed on FlexO-x multi-frame structure

Z	x	FlexO-x Rows × Columns	# OFC blocks	PAD (bits)	Pre FEC (bits)	Post FEC encode (bits)	Figure
8	**4**	116 × 10280	168	992	1193472	1376256	16-5
8	**2**	116 × 10280					16-6
4	**2**	58 × 10280	84	496	596736	688128	16-7
4	**1**	116 × 5140					16-8

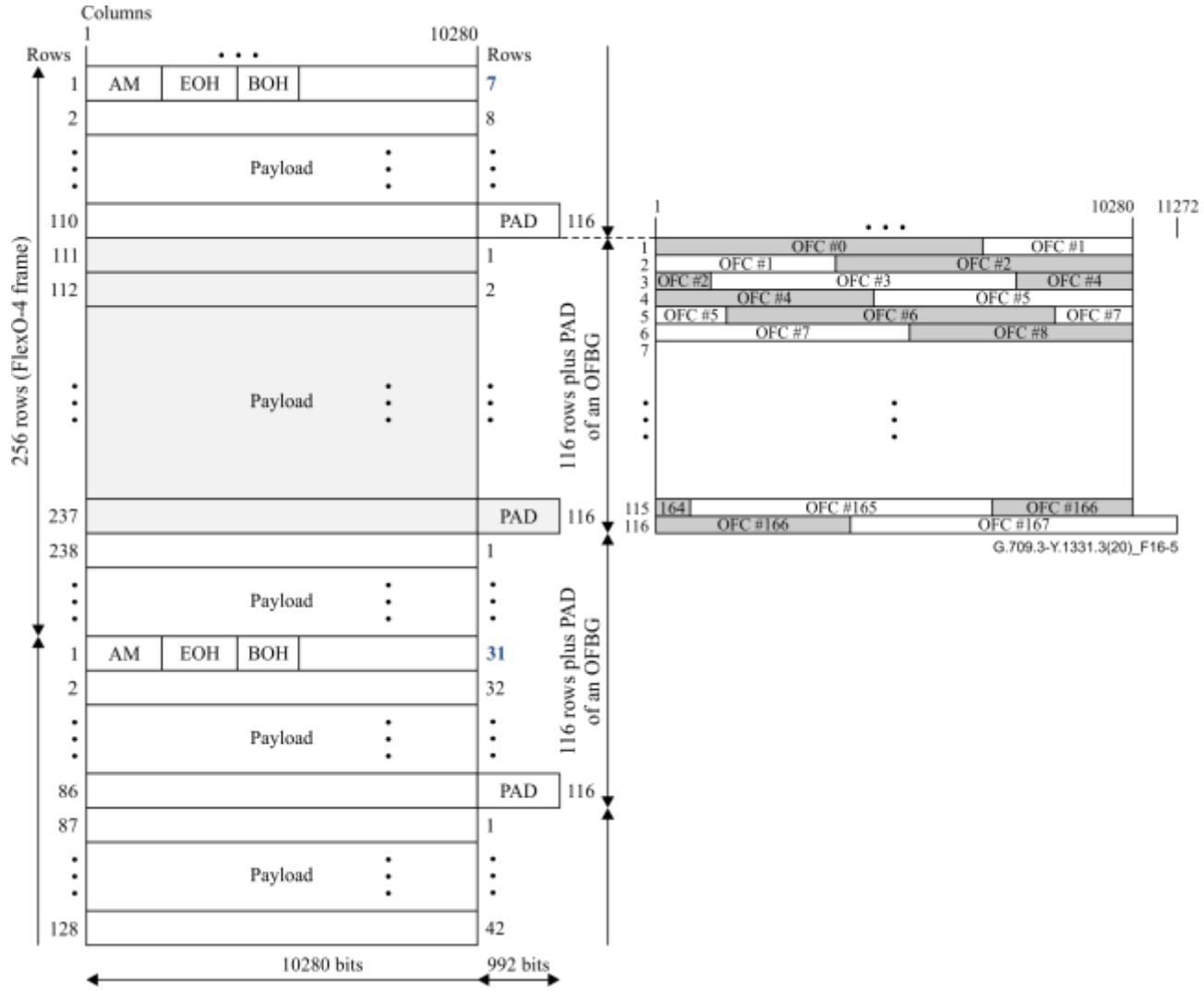


Figure 16-5 – OFBG8 superimposed on FlexO-4 multi-frame structure

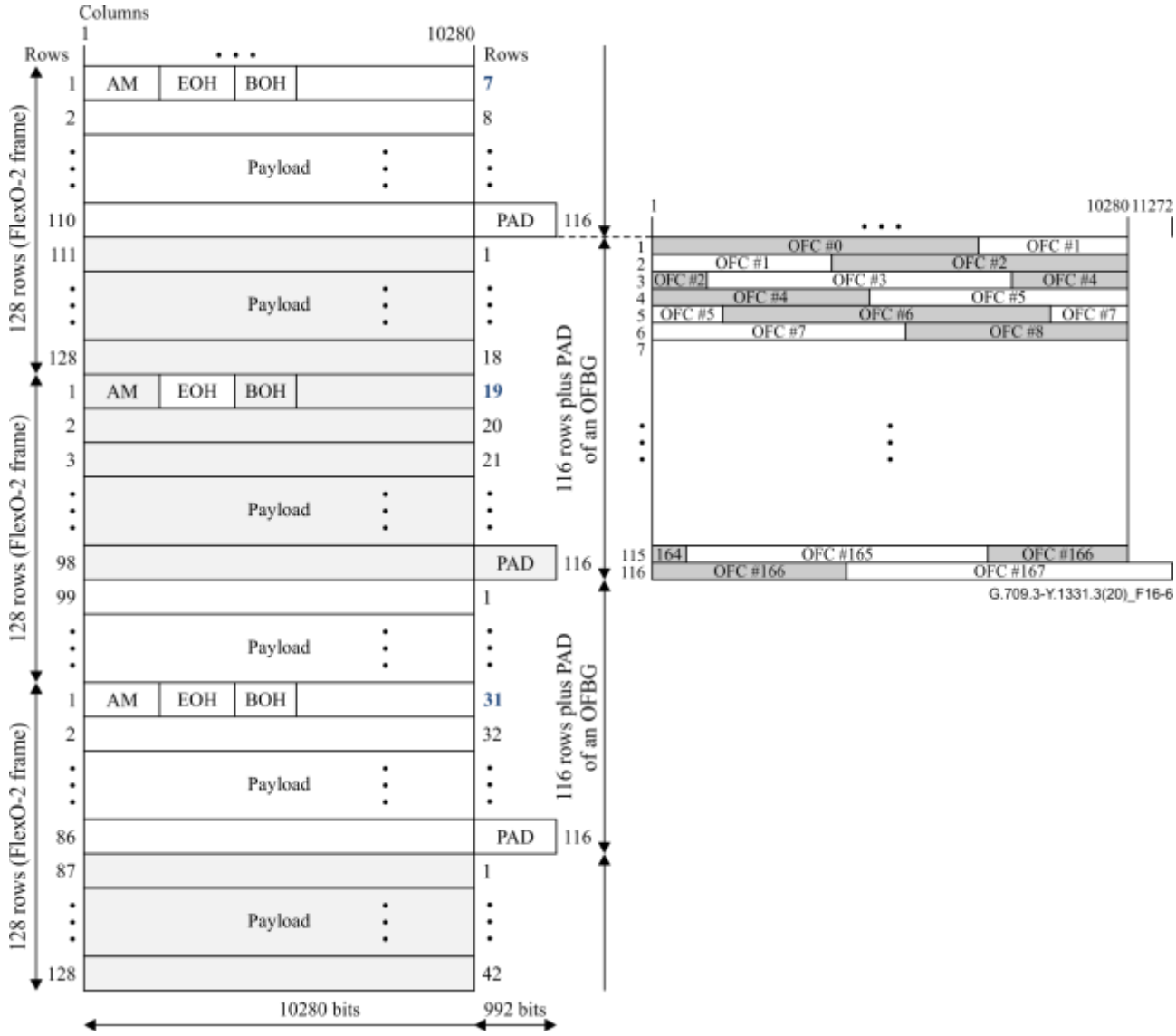


Figure 16-6 – OFBG8 superimposed on FlexO-2 multi-frame structure

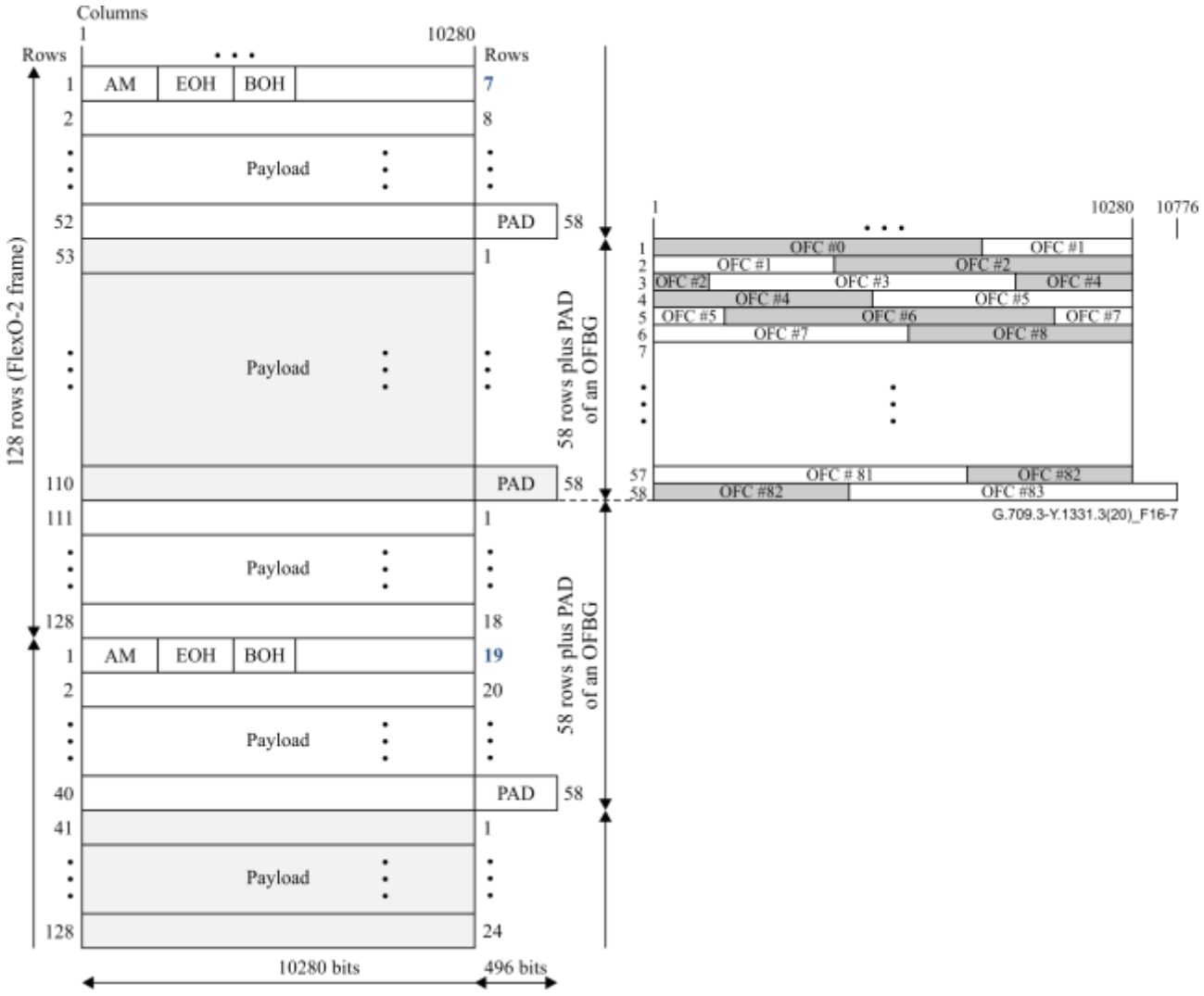


Figure 16-7 – OFBG4 superimposed on FlexO-2 multi-frame structure

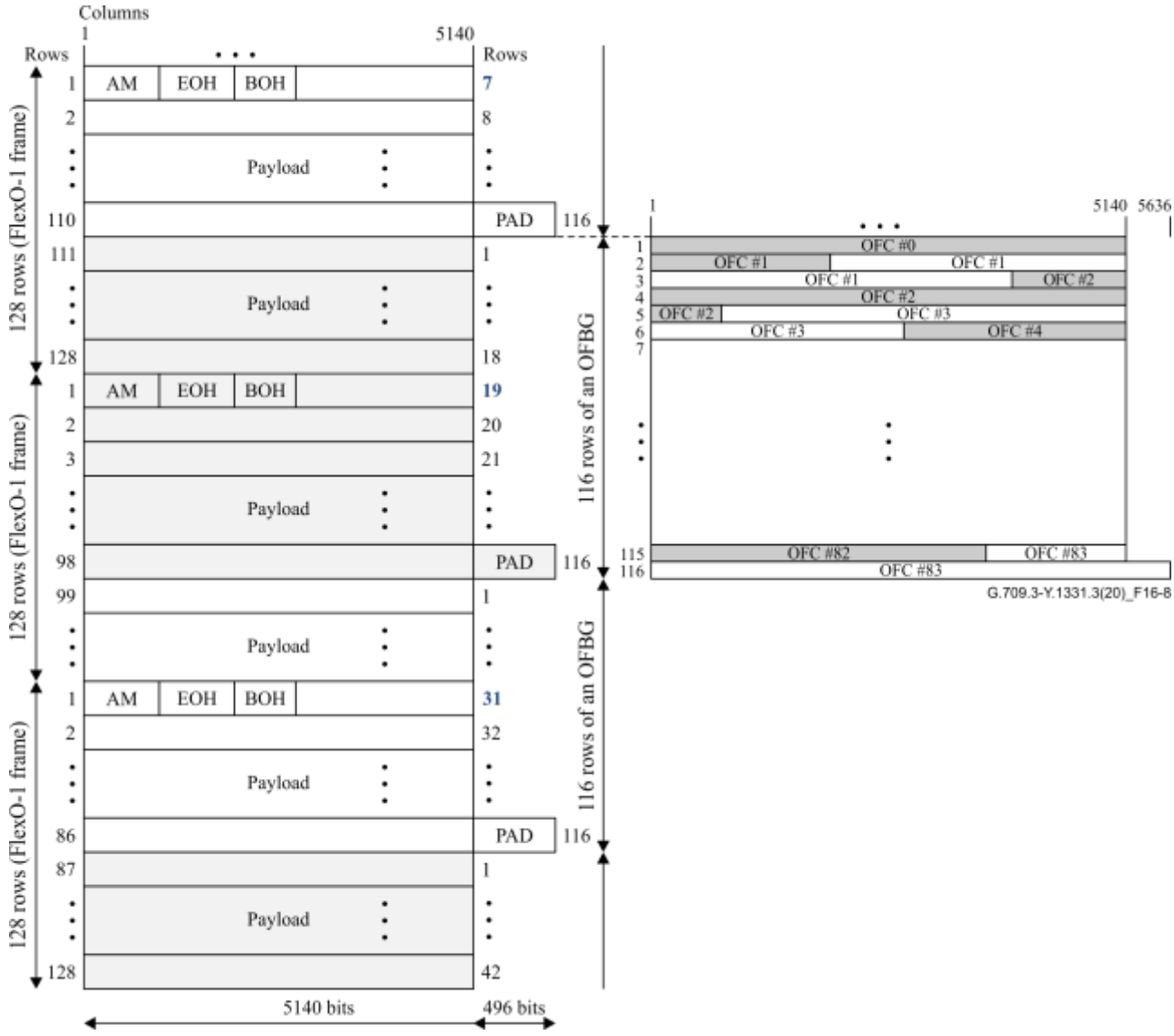


Figure 16-8 – OFBG4 superimposed on FlexO-1 multi-frame structure

16.4.2 OFBGz scrambling

The OFBGz is scrambled. The operation of the scrambler shall be functionally equivalent to that of a frame-synchronous scrambler of sequence 65535 and the generating polynomial shall be $x^{16} + x^{12} + x^{3} + x + 1$.

The scrambler resets to 0xFFFF on row 1, column 1 of the OFBG structure. The scrambler state advances during each bit of the OFBGz structure.

16.4.3 OFBGz structure representations

An OFBGz structure consists of 21×Z rows by 7104 1-bit columns. It is illustrated in the top of Figure 16-9. The rows are numbered from 0 to 21 × Z – 1. The columns are numbered from 0 to 7103. Each row represents one OFC block.

An OFBGz is split into two parts on a bit-by-bit basis. The even columns (i.e., #0, 2, 4, .., 7102) form a structure consisting of 21×Z rows by 3552 1-bit columns, and is identified as OFBGz.0 (see middle part of Figure 16-9). The odd columns (i.e., #1, 3, 5, .., 7103) form a structure consisting of 21×Z rows by 3552 1-bit columns, and is identified as OFBGz.1. Each row represents one OFCi (i = 0,1) block.

The first 3072 bits of an OFCi block are partitioned into 192 16-bit blocks, numbered 1 to 192. The last 480 bits of an OFCi block are partitioned into 32 15-bit blocks, numbered 193 to 224.

These 224 blocks of an OFCi are organized in a 32 row by 7 column matrix structure (see bottom of Figure 16-9), in which each row contains 111 bits. Column K = 0 contains 16-bit blocks 1 to 32, column K = 1 contains 16-bit blocks 33 to 64, etc. and column K = 6 contains 15-bit blocks 193 to 224. The 21×Z OFCi blocks in this matrix structure are identified as matrix Ui (i = 0,1).

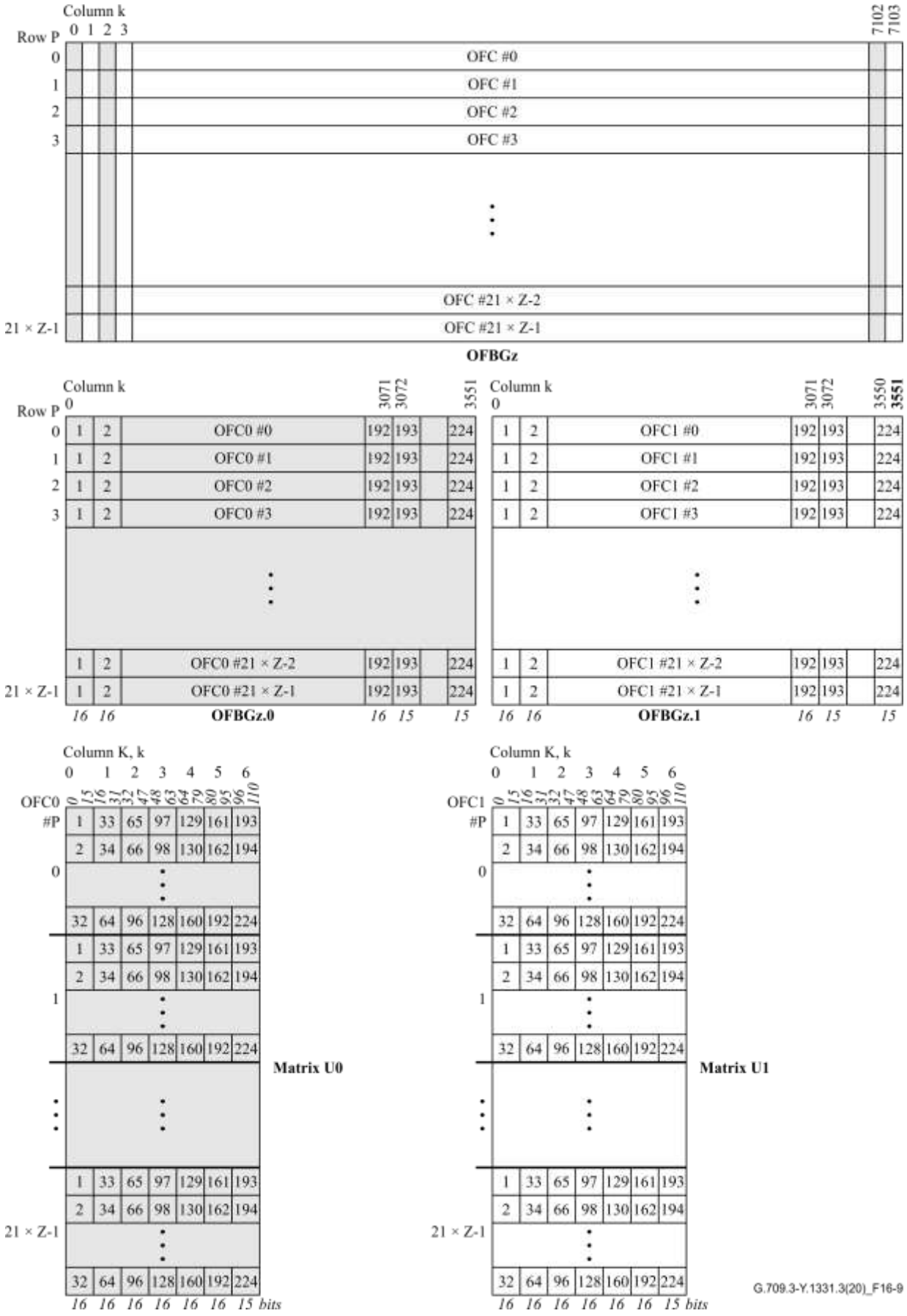


Figure 16-9 – Three OFBGz representations

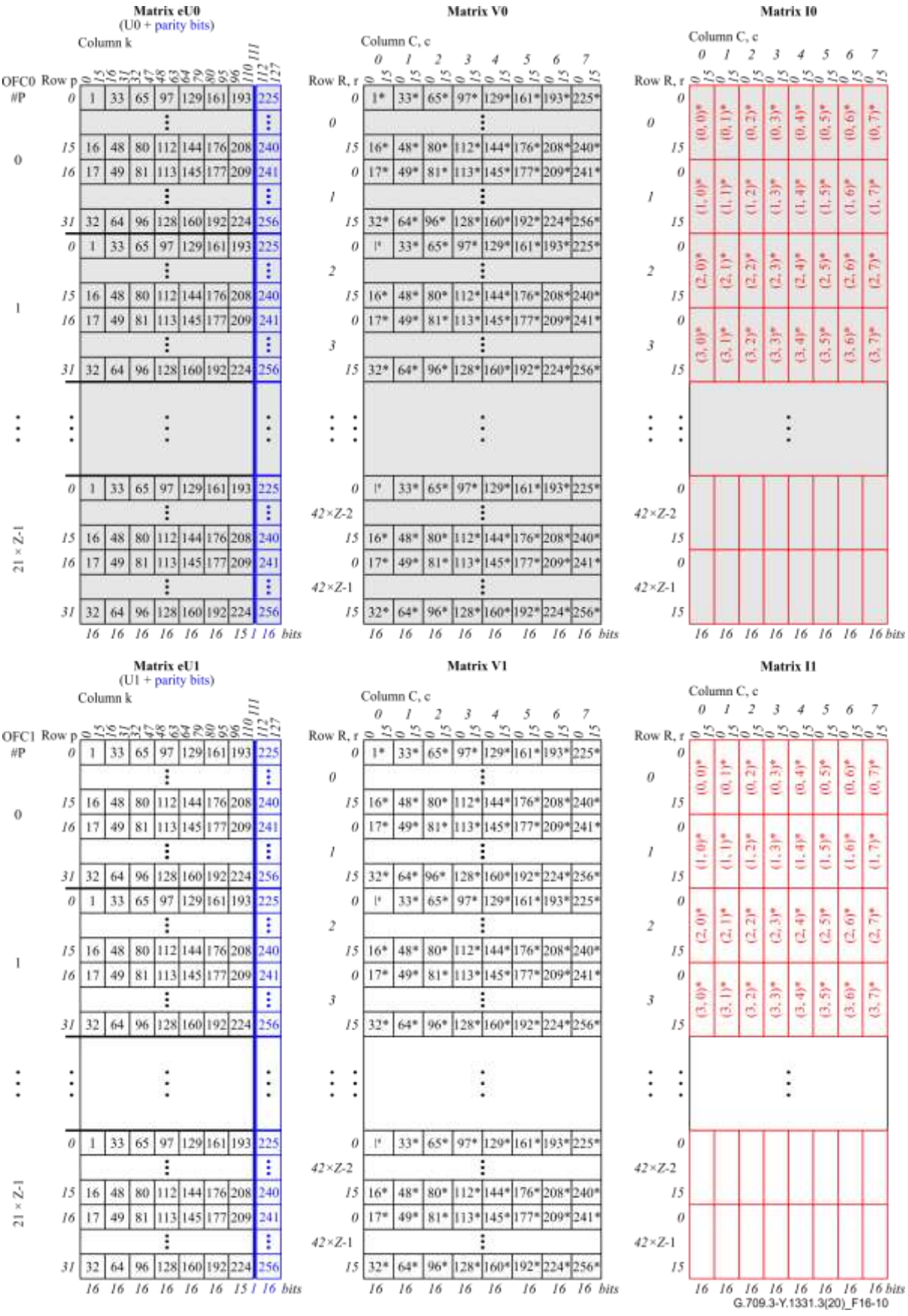


Figure 16-10 – OFBGz with parity, parity and row permutation (Vi, i=0,1) and parity and row and square block permutations (Ii, i=0,1)

16.4.4 Forward error correction

The FlexO-x-DO FEC code is based on an extended *BCH*(256,239) code as specified in Annex E that works in conjunction with an interleaver function. It adds 17-bits of parity to each of the (32 × 21×Z = 672×Z) 111-bit blocks in matrix Ui (i = 0,1), extending it to (672×Z) 128-bit blocks (see left side of Figure 16-10).

Each row of 128-bit in this extended matrix Ui (eUi) is identified by a 2-tuple row number {P,p} in which P represents an OFCi instance with its parity bits (OFCPi) in the range 0 to 21 × Z – 1 and p represents a row number with value in the range 0 to 31. Each column in this matrix eUi is identified by a column number k with a value in the range of 0 to 127. Each bit in this matrix eUi is identified by the 3-tuple {P,p,k}.

The eight 16-bit blocks in each 128-bit row are permuted (j ➔ j*, j = 1 to 256) and the result is identified as matrix Vi (i = 0,1) (see middle of Figure 16-10).

Each row in this matrix Vi is identified by a 2-tuple row number {R,r} in which R represents a value in the range 0 to 42 × Z – 1, and r represents a value in the range 0 to 15. Each column in this matrix Vi is identified by a 2-tuple column number {C,c} in which C represents a value in the range of 0 to 7 and c represents a value in the range 0 to 15. Each bit in this matrix Vi is identified by the 4-tuple {R,C,r,c}.

After FEC parity computation and addition of the 17 parity bits to matrix Ui, the bits {P,p,k} of matrix eUi map into bits {R,C,r,c} of matrix Vi as specified in equations 16-1a and 16-1b with ranges of P, p, k, R, C, r and c as specified in Table 16-6.

Equation 16-1a specifies the mapping from the perspective of bit {R,C,r,c} in matrix Vi. Equation 16-1b specifies the mapping from the perspective of bit {P,p,k} in matrix eUi.

$$Vi\{R,C,r,c\} = eUi\{P = \lfloor R/2 \rfloor,\ p = (R\ \%\ 2) \times 16 + r,\ k = 16 \times C + (r\ \hat{}\ c)\} \quad (16\text{-}1a)$$

$$Vi\{R = 2P + \lfloor p/16 \rfloor,\ C = \lfloor k/16 \rfloor,\ r = p\ \%\ 16,\ c = (k\ \%\ 16)\ \hat{}\ (p\ \%\ 16)\} = eUi\{P,p,k\} \quad (16\text{-}1b)$$

NOTE 1 – Refer to clause IV.2 for an illustration of equation 16-1.

Table 16-6 – Ranges of P, p, k, R, r, C and c and number of rows in eUi and Vi

Z	Range of P	Range of p	Range of k	Range of R	Range of r	Range of C	Range of c	# of rows in Ui, Vi
8	0 .. 167	0 .. 31	0 .. 127	0 .. 335	0 .. 15	0 .. 7	0 .. 15	5376
4	0 .. 83	0 .. 31	0 .. 127	0 .. 167	0 .. 15	0 .. 7	0 .. 15	2688

The 17 parity bits Wi(P,p,k, k = 239..255) of an extended *BCH*(256,239) code word Wi(P,p), with P $\in$ {0..21×Z-1} and p $\in$ {0..31}, are computed – according the specification in Annex E – over 128 bits from matrix Vi and 111 bits from matrix Ui. Encoding is done sequentially, in order of increasing rows. At the time when a constituent code word (P, p) is being encoded, all constituent codes (P', p') with P' < P – 2 must already be encoded.

Note that the parameter "t" is added to Wi and Vi in equation 16-2 to address the behaviour at the boundary of two consecutive matrices Vi{t-1} and Vi{t}.

Equations 16-2 and 16-3 specify which Vi{R,C,r,c} and Ui{P,p,k} bits are used in a Wi(P,p):

$$Wi(t,P,p,k,\ k = 0..127) = Vi\ \{\ \text{if}\ (\ (2P + \lfloor p/16 \rfloor)\ \hat{}\ 1) - 20 + 2 \times \lfloor k/16 \rfloor < 0\ \text{then t-1 else t}\ ,\ (\ (\ (2P + \lfloor p/16 \rfloor)\ \hat{}\ 1) - 20 + 2 \times \lfloor k/16 \rfloor)\ \%\ (42 \times Z)\ ,\ \lfloor k/16 \rfloor\ ,\ (k\ \%\ 16)\ \hat{}\ (p\ \%\ 16)\ ,\ (p\ \%\ 16)\ \} \quad (16\text{-}2)$$

$$Wi(P,p,k,\ k = 128..238) = Ui\ \{\ P,\ p,\ k - 128\ \} \quad (16\text{-}3)$$

NOTE 2 – Wi(t , *P < 10* , p, k) with k < 128 results in a case that all 128 bits or a subset of 128 bits have Vi{*t-1*}. Vi{*t-1*} values refer to bits in the previous instance of Vi. For example, Vi(t){R = -1 , C , r , c} refers to Vi(t-1){R = 335 , C , r , c} for Z = 8 and to Vi(t-1){R = 167 , C , r , c} for Z = 4.

Figure 16-11 provides a graphical illustration of the Ui{P,p,k} and Vi{t,R,C,r,c} bits that contribute to Wi(t,P,p,k) (i = 0,1) for p = 20 (see red elements) and p = 11 (see blue elements) and any value of P.

– For the case of p = 20, the 16 bits in column c = (p % 16) = 4 of Vi{t,R,C} with {R,C} = {20P-20,0}, {20P-18,1}, {20P-16,2}, {20P-14,3}, {20P-12,4}, {20P-10,5}, {20P-8,6} and {20P-6,7} are mapped into the first eight 16-bit blocks of Wi(t , P , p=20).

– For the case of p = 11, the 16 bits in column c = (p % 16) = 11 of Vi{t,R,C} with {R,C} = {20P-19,0}, {20P-17,1}, {20P-15,2}, {20P-13,3}, {20P-11,4}, {20P-9,5}, {20P-7,6} and {20P-5,7}are mapped into first eight 16-bit blocks of Wi(t , P , p=11).

– The mapping is performed after permutation of the 16 bits in each column c of these Vi{t,R,C}. The first bit after permutation of a 16-bit column is then mapped into the first bit of the associated 16-bit block in Wi(t,P,p), the second bit after permutation of the 16-bit column is mapped into the second bit of the associated 16-bit block in Wi(t,P,p), etc. as indicated in Figure 16-11 and Table IV.2.

NOTE 3 – The permutation of the 16 bits in each column c in a Vi{t,R,C} under control of the term ((k % 16) ^ (p % 16)) is the same as the permutation of the 16 bits in a 16-bit block in a row in matrix eUi. Refer to Appendix IV.2 for an illustration.

NOTE 4 – Refer to clause IV.3 for an illustration of equations 16-2 and 16-3.

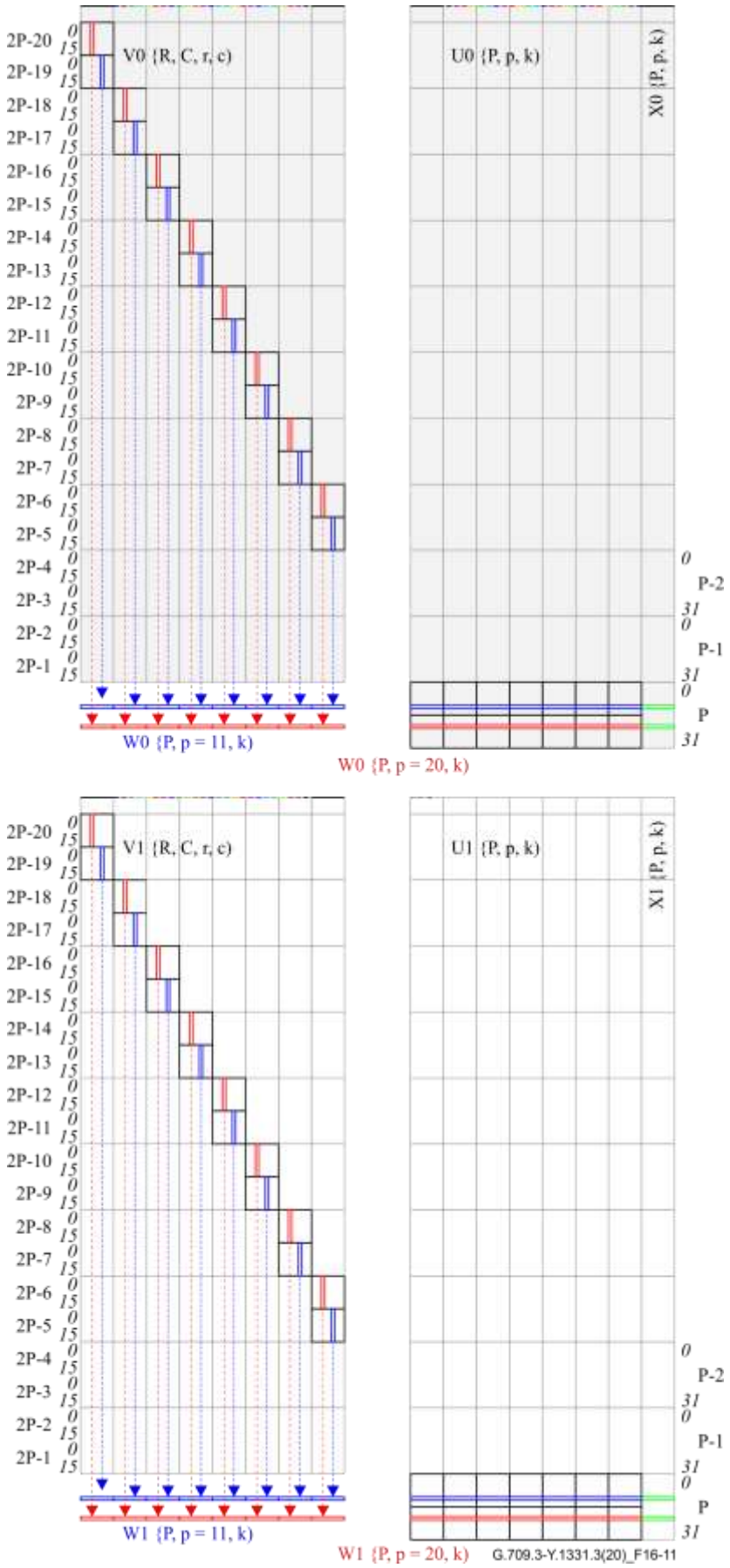


Figure 16-11 – Illustration of Wi(P,p) for p = 11 (blue) and 20 (red)

16.4.5 Interleaving

The interleaver process consists of two stages. The first stage provides an intra-block interleaving, the second stage provides an inter-block interleaving.

The intra-block interleaver reorders the bits in each 16×16 block of Vi to ensure that the bits in each row and column of a block at the encoder output are remapped almost uniformly in the block for transmission on the line.

The inter-block interleaver attempts to have nearby symbols on the line contain bits that are widely separated in the encoder output. Bits are therefore read by columns, rather than rows because interleaver columns are much longer than rows, so bits in a column are spread over more constituent codes than bits in a row, which increases the tolerance to long bursts.

16.4.5.1 Intra-block interleaving

For the intra-block interleaving, the 16 × 16 bits in a Vi{R,C} block are permuted ({R,C} ➔ {R,C}*, R = 0 to 42 × Z – 1 and C = 0 to 7), and the result is identified as matrix Ii (i = 0,1) (see right-hand side of Figure 16-10).

Each row in this matrix Ii is identified by a 2-tuple row number {R,r} in which R represents a value in the range 0 to 42 × Z – 1 and r represents a value in the range 0 to 15. Each column in this matrix Ii is identified by a 2-tuple column number {C,c} in which C represents a value in the range of 0 to 7 and c represents a value in the range 0 to 15. Each bit in this extended matrix Ui is identified by the 4-tuple {R,C,r,c}.

The bits {R,C,r,c} of matrix Vi map into bits {R,C,r,c} of matrix Ii as specified in equation 16-4 with ranges of R, C, r and c as specified in Table 16-6.

$$Ii\{R,C,r,c\} = Vi\{R, C, ((c - 2r - \lfloor r/8 \rfloor) \% 16), ((c - r - \lfloor r/8 \rfloor) \% 16)\} \quad (16\text{-}4)$$

NOTE – Refer to clause IV.4 for an illustration of equation 16-4.

16.4.5.2 Inter-block interleaving

The inter-block interleaving is specified in 16.4.6 as part of the OFBGz mapping into the payload and FEC parity area of the FlexO-x-DO multi-frame.

16.4.6 Mapping of OFBGz with Parity (OFBGPz) into FlexO-x-DO (Z)

For this mapping purpose, matrices I0 and I1 are divided into inter-block interleaving sets of 21 rows of eight 16×16-bit blocks {R, C=0..7}. Two of such inter-block interleaving sets in both I0 and I1 form a super-set of four inter-block interleaving sets, which are numbered 0 to 3. Inter-block interleaving sets 0 and 2 are in I0 and inter-block interleaving sets 1 and 3 are in I1. A super-set contains (4 × 21 =) 84 rows of eight 16×16-bit blocks. Matrices I0 and I1 contain a total of Z super-sets, numbered 0 to Z-1.

Table 16-7 – Inter-block interleaving set and super-set numbers

Z	# super-sets	# inter-block interleaving sets per super-set	# bits per super-set	# Z-bit blocks per super-set
8	8	4	172032	21504
4	4	4	172032	43008

The OFBGPz mapping is performed on super-set by super-set basis in the order 0, 1, 2, .., Z-1 (top-to-bottom) and on 8 column bits by 8 column bits basis within each super-set in the order 8 column bits from inter-block interleaving set #0, 8 column bits from inter-block interleaving set #1, 8 column bits from inter-block interleaving set #2, 8 column bits from inter-block interleaving set #3,

8 column bits from inter-block interleaving set #0, 8 column bits from inter-block interleaving set #1, etc. Within each inter-block interleaving set, the first 42 8-bit blocks are from column {C,c} = {0,0}, followed by 42 8-bit blocks from column {0,1}, etc. until 42 8-bit blocks from column {7,15}; i.e., top-to-bottom, left-to-right in Figure 16-13.

NOTE – Refer to clause IV.5 for some worked out examples.

16.4.6.1 Mapping of OFBGP8 into FlexO-x-DO (Z=8) (DP-16QAM symbols)

Figure 16-12 and Table 16-8 specify the mapping of 8-bit blocks from matrices I0 and I1 into the payload and FEC parity area of the FlexO-x-DO (Z=8) multi-frame.

Table 16-8 – 8-bit block mapping into FlexO-x-DO Z-bit blocks (Z = 8)

FlexO-x-DO (Z=8)			8-bit block
Frame	Row	Column	
1	4	15	I0{0,0,0-7,0}
1	4	16	I1{0,0,0-7,0}
1	4	17	I0{21,0,0-7,0}
1	4	18	I1{21,0,0-7,0}
1	4	19	I0{0,0,8-15,0}
1	4	20	I1{0,0,8-15,0}
:	:	:	:
7	3	30	I0{20,7,8-15,7}
7	3	31	I1{20,7,8-15,7}
7	3	32	I0{41,7,8-15,7}
7	4	2	I1{41,7,8-15,7}
7	4	3	I0{42,0,0-7,0}
7	4	4	I1{42,0,0-7,0}
:	:	:	:
48	116	29	I0{314,7,8-15,7}
48	116	30	I1{314,7,8-15,7}
48	116	31	I0{335,7,8-15,7}
48	116	32	I1{335,7,8-15,7}

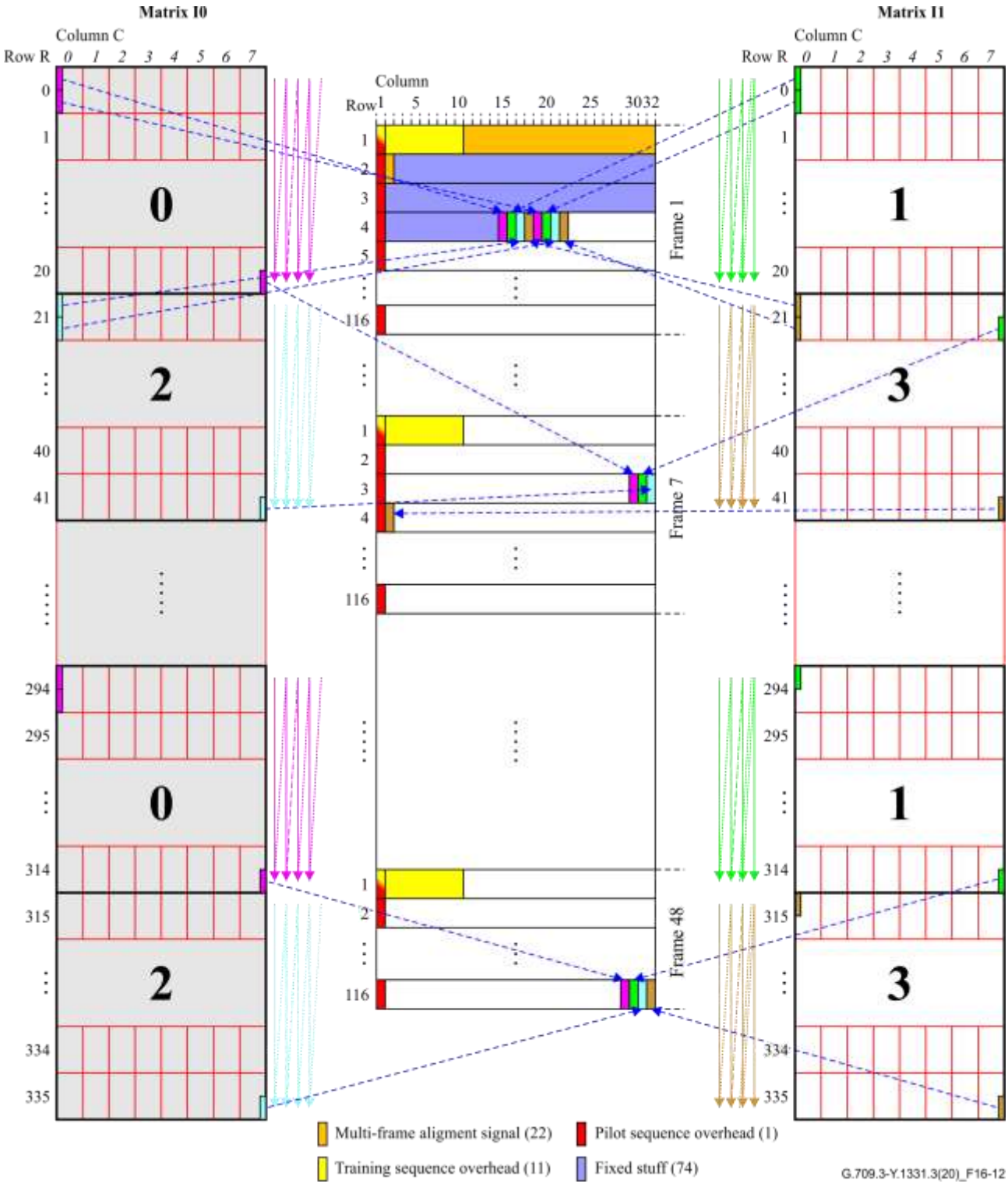


Figure 16-12 – Mapping of OFBGP8 into FlexO-x-DO (x = 2,4)

16.4.6.2 Mapping of OFBGP4 into FlexO-x-DO (Z=4) (DP-QPSK symbols)

Figure 16-13 and Table 16-9 specify the mapping of 8-bit blocks from matrices I0 and I1 into the payload and FEC parity area of the FlexO-x-DO (Z=4) multi-frame.

Table 16-9 – 8-bit block mapping into FlexO-x-DO Z-bit blocks (Z = 4)

FlexO-x-DO (Z=4)			8-bit block
Frame	**Row**	**Columns**	
1	4	15,16	I0{0,0,0-7,0}
1	4	17,18	I1{0,0,0-7,0}
1	4	19,20	I0{21,0,0-7,0}
1	4	20,21	I1{21,0,0-7,0}
1	4	22,23	I0{0,0,8-15,0}
1	4	24,25	I1{0,0,8-15,0}
:	:	:	:
13	2	9,10	I0{20,7,8-15,7}
13	2	11,12	I1{20,7,8-15,7}
13	2	13,14	I0{41,7,8-15,7}
13	2	15,16	I1{41,7,8-15,7}
13	2	17,18	I0{42,0,0-7,0}
13	2	19,20	I1{42,0,0-7,0}
:	:	:	:
48	116	25,26	I0{146,7,8-15,7}
48	116	27,28	I1{146,7,8-15,7}
48	116	29,30	I0{167,7,8-15,7}
48	116	31,32	I1{167,7,8-15,7}

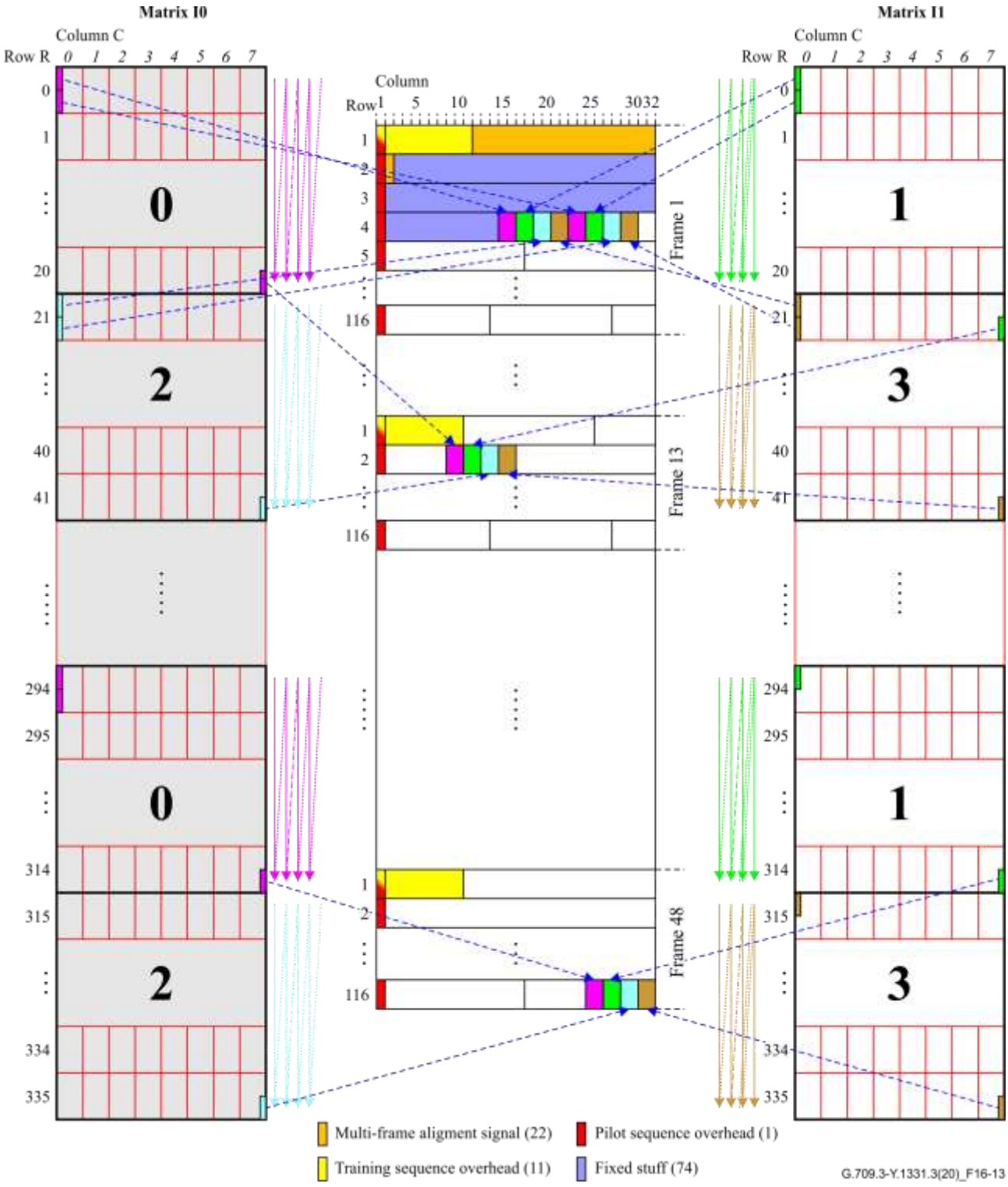


Figure 16-13 – Mapping of OFBGP4 into FlexO-x-DO (x = 1,2)

16.5 FOICx.k-DO

A conceptually serial FlexO-x-DO signal is adapted to a parallel multi-lane distribution (MLD) signal format with k lanes, referred to as FOICx.k-DO.

16.5.1 FOIC1.4-DO lanes

The FlexO-1-DO (Z=4) bits are distributed to four logical FOIC1.4-DO lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of

every 4-bit block is carried on lane #i (i = 1..4). Each FOIC1.4-DO lane is synchronous to the FlexO-1-DO (Z=4) frame.

Each FlexO-1-DO (Z=4) multi-frame contains 3712 × 48 × 4 = 712704 bits. Each FOIC1.4-DO lane will carry 25% of these bits, which are 3712 × 48 = 178176 bits.

The bit rates and tolerance of the FOIC1.4-DO lanes are defined in Table 16-10.

Table 16-10 – FOIC1.4-DO types and bit rates

FOIC1.4-DO lane nominal bit rate	Bit-rate tolerance
2936832/2314805 × 24 883200 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC1.4-DO lane bit rates is approximately: 31 569 733.961 kbit/s.	

16.5.2 FOIC2.4-DO lanes

The FlexO-2-DO (Z=4) bits are distributed to four logical FOIC2.4-DO lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 4-bit block is carried on lane #i (i = 1..4). Each FOIC2.4-DO lane is synchronous to the FlexO-2-DO (Z=4) frame.

Each FlexO-2-DO (Z=4) multi-frame contains 3712 × 48 × 4 = 712704 bits. Each FOIC2.4-DO lane will carry 25% of these bits, which are 3712 × 48 = 178176 bits.

The bit rates and tolerance of the FOIC2.4-DO lanes are defined in Table 16-11.

Table 16-11 – FOIC2.4-DO types and bit rates

FOIC2.4-DO lane nominal bit rate	Bit-rate tolerance
2936832/2314805 × 49 766 400 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC2.4-DO lane bit rates is approximately: 63 139 467.923 kbit/s.	

16.5.3 FOIC2.8-DO lanes

The FlexO-2-DO (Z=8) bits are distributed to eight logical FOIC2.8-DO lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 8-bit block is carried on lane #i (i = 1..8). Each FOIC2.8-DO lane is synchronous to the FlexO-2-DO (Z=8) frame.

Each FlexO-2-DO (Z=8) multi-frame contains 3712 × 48 × 8 = 14254408 bits. Each FOIC2.8-DO lane will carry 12.5% of these bits, which are 3712 × 48 = 178176 bits.

The bit rates and tolerance of the FOIC2.8-DO lanes are defined in Table 16-12.

Table 16-12 – FOIC2.8-DO types and bit rates

FOIC2.8-DO lane nominal bit rate	Bit-rate tolerance
2936832/2314805 × 24 883 200 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC2.8-DO lane bit rate is approximately: 31 569 733.961 kbit/s.	

16.5.4 FOIC4.8-DO lanes

The FlexO-4-DO (Z=8) bits are distributed to eight logical FOIC4.8-DO lanes, on a bit-by-bit basis, in a round robin distribution scheme from the lowest to the highest numbered lanes so that bit #i of every 8-bit block is carried on lane #i (i = 1..8). Each FOIC4.8-DO lane is synchronous to the FlexO-4-DO (Z=8) frame.

Each FlexO-4-DO (Z=8) super-frame contains 3712 × 48 × 8 = 1425408 bits. Each FOIC4.8-DO lane will carry 12.5% of these bits, which are 3712 × 48 = 178176 bits.

The bit rates and tolerance of the FOIC4.8-DO lanes are defined in Table 16-13.

Table 16-13 – FOIC4.8-DO types and bit rates

FOIC4.8-DO lane nominal bit rate	Bit-rate tolerance
2936832/2314805 × 49 766 400 kbit/s	±20 ppm
NOTE 1 – The nominal FOIC4.8-DO lane bit rate is approximately: 63 139 467.923 kbit/s.	

Annex A

Forward error correction using 512 × 510 staircase codes

(This annex forms an integral part of this Recommendation.)

Refer to Annex A of [ITU-T G.709.2].

Annex B

Adaptation of 512 × 510 staircase codes to 100G FlexO-1-SC FEC

(This annex forms an integral part of this Recommendation.)

Annex A.2 of [ITU-T G.709.2] describes a generic base block that is used to create the 512 × 510 bit staircase block.

For generating staircase FEC code words of a 100G FlexO-1-SC signal, this base block will be created by mapping the 100G FlexO-1-SC information and FEC bits into it.

Annex B details these mapping specific aspects.

B.1 100G FlexO-1-SC bit and SC FEC specific base blocks mapping relationship

The staircase FEC scheme (with error de-correlator) is specified to operate on base blocks, which contain (8×30592) FEC information bits and (8×2048) FEC parity bits.

The bits of a FBG map into five base blocks. The boundaries of the five information and parity blocks within an FBG are illustrated in Figure B.1.

To compute the FEC parity, the FBG bits in columns 1 to 5140 and the FlexO-1-SC overhead bits in columns 5141 to 5485 are mapped into the first 30952 columns of five consecutive base blocks as illustrated in Figure B.1.

The computed FEC parity bits located in the last 2048 columns of five consecutive base blocks are mapped into 16384 (out of 16422) parity bits in columns 5141 to 5485 of a FBG.

The total number of information bits in a FBG is:

$$238\times5140 \text{ bits} = 1{,}223{,}320 \text{ bit} = 5\times244{,}664 = 5\times [244{,}736 - 72] = 5\times [(512\times478) - 72]$$

The 244,664 information bits of B_{i-1} in FBG row j+1, column 1 to row j+48, column 3084 (see Figure B.1) are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 (i.e., 30591-72) of Base block 1.

The 244,664 information bits of B_i in FBG row j+48, column 3085 to row j+96, column 1028 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the Base block 2.

The 244,664 payload information bits of B_{i+1} in FBG row j+96, column 1029 to row j+143, column 4112 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the Base block 3.

The 244,664 payload information bits of B_{i+2} in FBG row j+143, column 4113 to row j+191, column 2056 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the Base block 4.

The 244,664 payload information bits of B_{i+3} in FBG row j+191, column 2057 to row j+238, column 5140 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the Base block 5.

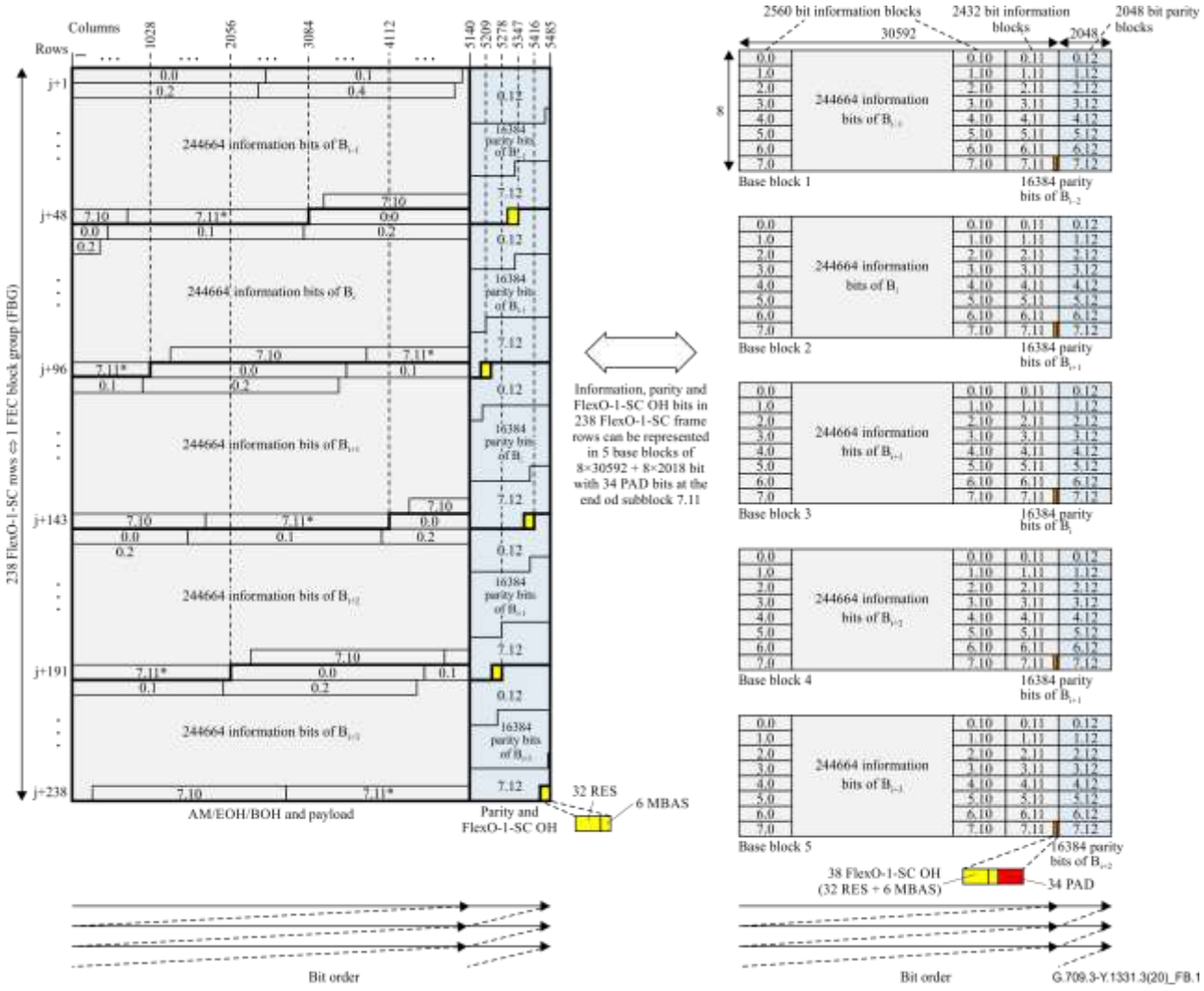


Figure B.1 – 100G FlexO-1-SC bit and SC FEC specific Base Blocks mapping relationship

The 38 FlexO-1-SC overhead bits in FBG row j+48, columns 5310 to 5347 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the Base block 1.

The 38 FlexO-1-SC overhead bits in FBG row j+96, columns 5172 to 5209 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the Base block 2.

The 38 FlexO-1-SC overhead bits in FBG row j+143, columns 5379 to 5416 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the Base block 3.

The 38 FlexO-1-SC overhead bits in FBG row j+191, columns 5241 to 5278 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the Base block 4.

The 38 FlexO-1-SC overhead bits in FBG row j+238, columns 5448 to 5485 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the Base block 5.

The last 34 bits in row 7, columns 30558 to 30591 of the payload information area of Base blocks 1 to 5 are always assumed with an all-zero value for the FEC parity calculations and not transported by FlexO-1-SC.

The total number of FEC parity bits in the 238 FlexO-1-SC frame row sequence is:

$$238\times345 = 82{,}110 \text{ bit} = 5\times16{,}422 = 5\times [38 + 16{,}384] = 5\times [38 + (512\times32)]$$

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of Base block 1 are mapped to the 16,384 parity bits of B_{i-2} in FBG row j+1, column 5141 to row j+48, column 5309 (see Figure B.1).

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of Base block 2 are mapped to the 16,384 parity bits of B_{i-1} in FBG row j+48, column 5348 to row j+96, column 5171.

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of Base block 3 are mapped to the 16,384 parity bits of B_i in FBG row j+96, column 5210 to row j+143, column 5378.

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of Base block 4 are mapped to the 16,384 parity bits of B_{i+1} in FBG row j+143, column 5417 to row j+191, column 5240.

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of Base block 5 are mapped to the 16,384 parity bits of B_{i+2} in FBG row j+191 column 5279 to row j+238, column 5447.

The bit order of the information bits in columns 1 to 5140 of the FlexO-1-SC frame and the bit order of information bits of the 8×30592 bit information base blocks is the same, and is from left-to-right and top-to-bottom.

The bit order of the parity bits in columns 5141 to 5485 of the FlexO-1-SC frame and the bit order of parity bits of the 8×2048 bit parity base blocks is the same, and is from left-to-right and top-to-bottom.

B.2 100G FlexO-1-SC transmitter and receiver SC FEC processing

Figure B.2 presents a 100G FlexO-1-SC specific version of Figure A.1 from [ITU-T G.709.2]. In Figure B.2, the "Input frame without FEC parity", "Output frame with SC FEC parity" and "Output frame without FEC parity" blocks within Figure A.1 of [ITU-T G.709.2] are replaced by a "100G FlexO-1" block that contains 244,664 information bits, a "FlexO-1-SC" block that contains 244,664 information, 16384 parity and 38 overhead bits and a "100G FlexO-1' block that contains 244,664 information bits, respectively. Furthermore, it is illustrated that the first 30592 columns of a base block in Figure B.2 contain "32 RES, 6 MBAS and 34 PAD" bits instead of "72 PAD" bits.

A vector of 244,664 information bits from a FBG is distributed and mapped into the first 30592 columns of a base block together with 38 overhead (32 RES + 6 MBAS) bits and 34 PAD bits. Then the staircase FEC specific transmit side processing is performed and 16,384 parity bits become available in the last 2048 columns of a base block. FlexO-1-SC information and overhead (32 RES + 6 MBAS) bits plus computed parity bits are then mapped into the 100G FlexO-1-SC frame format.

At the receive side the information, overhead and parity bits in the FlexO-1-SC signal are mapped into information and parity areas of base blocks. Then the staircase specific receive side processing is performed and decoded information bits are stored in the first 30592 columns of a base block. The information bits are then mapped to the 100G FlexO-1 frame format.

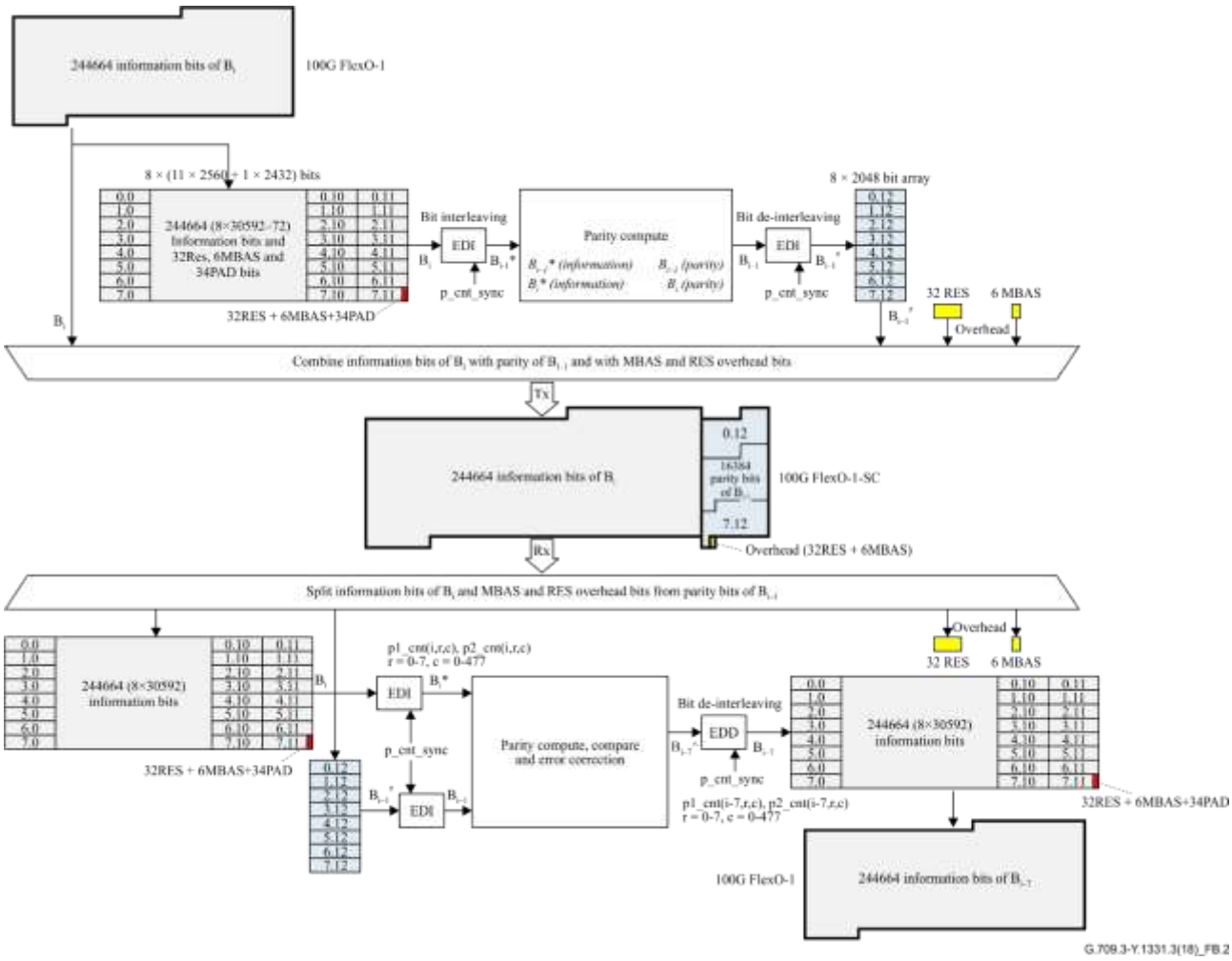


Figure B.2 – 100G FlexO-1-SC transmitter and receiver SC FEC processing

Annex C

Adaptation of 512 × 510 staircase codes to 200G|400G FlexO-x-SC FEC

(This annex forms an integral part of this Recommendation.)

Clause A.2 of [ITU-T G.709.2] describes a generic base block that is used to create the 512 × 510 bit staircase block.

For generating staircase FEC code words of a 200G|400G FlexO-x-SC signal (x = 2,4), this base block will be created by mapping the 200G|400G FlexO-x-SC information and FEC bits into it.

Annex C details these mapping specific aspects.

C.1 200G|400G FlexO-x-SC bit and SC FEC specific base blocks mapping relationship

The staircase FEC scheme (with error de-correlator) is specified to operate on base blocks, which contain (8×30592) FEC information bits and (8×2048) FEC parity bits.

The bits of a FBG map into five base blocks. The boundaries of the five information and parity blocks within an FBG are illustrated in Figure C.1.

To compute the FEC parity, the FBG bits in columns 1 to 10280 and the FlexO-x-SC overhead bits in columns 10281 to 10970 are mapped into the first 30952 columns of five consecutive base blocks as illustrated in Figure C.1.

The computed FEC parity bits located in the last 2048 columns of five consecutive base blocks are mapped into 16384 (out of 16422) parity bits in columns 10281 to 10970 of a FBG.

The total number of information bits in the 119 FlexO frame row sequence is:

$$119\times10280 \text{ bits} = 1{,}223{,}320 \text{ bit} = 5\times244{,}664 = 5\times [244{,}736 - 72] = 5\times [(512\times478) - 72]$$

The 244,664 information bits of B_{i-1} in FBG row j+1, column 1 to row j+24, column 8224 (see Figure C.1) are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 (i.e., 30591-72) of base block 1.

The 244,664 information bits of B_i in FBG row j+24, column 8225 to row j+48, column 6184 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the base block 2.

The 244,664 payload information bits of B_{i+1} in FBG row j+48, column 6185 to row j+72, column 4112 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the base block 3.

The 244,664 payload information bits of B_{i+2} in FBG row j+72, column 4113 to row j+96, column 2056 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the base block 4.

The 244,664 payload information bits of B_{i+3} in FBG row j+96, column 2057 to row j+119, column 10280 are mapped to the payload information bits in row 0, column 0 to row 7, column 30519 of the base block 5.

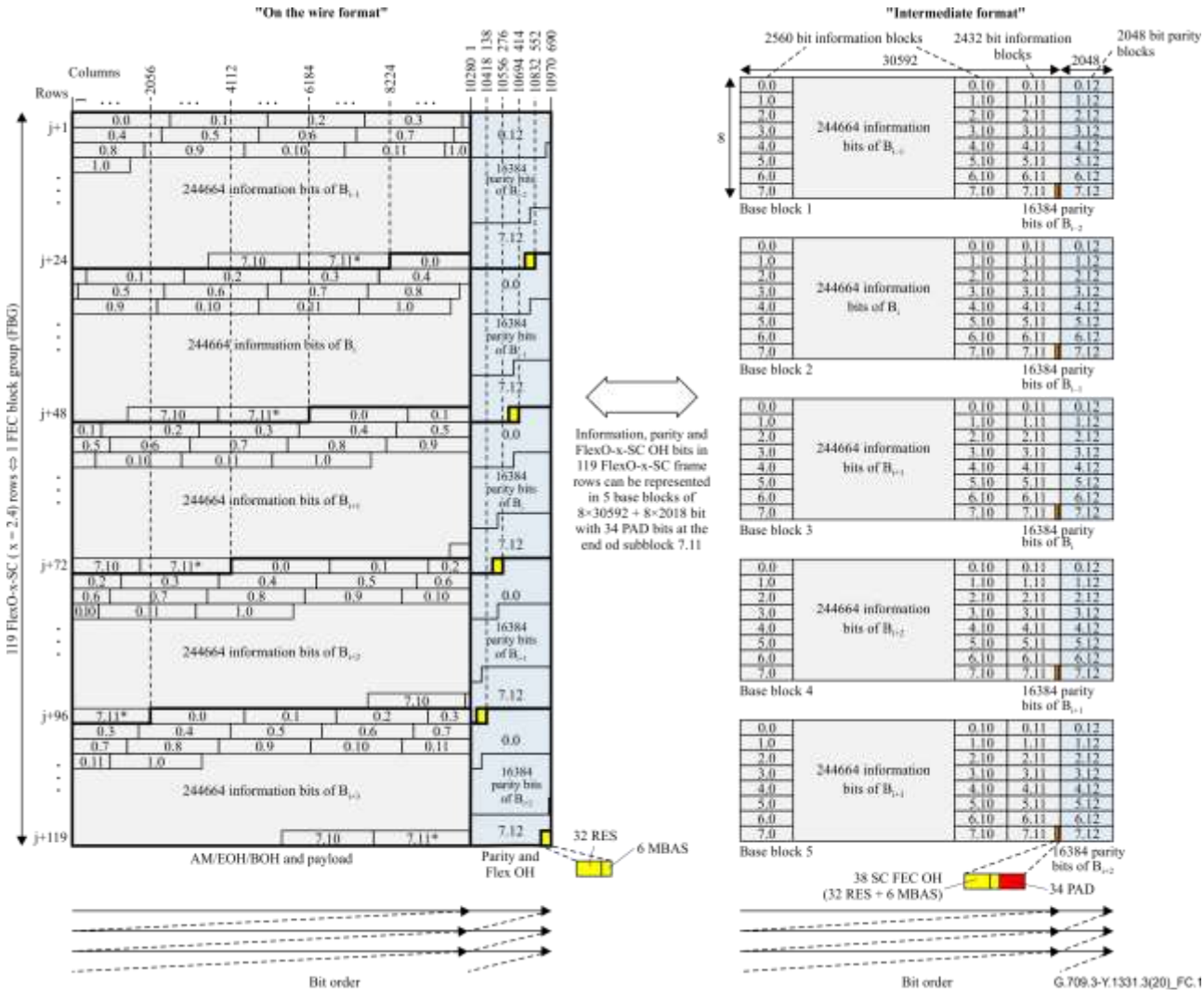


Figure C.1 – 200G|400G FlexO-x-SC bit and SC FEC specific base blocks mapping relationship

The 38 FlexO-x-SC overhead bits in FBG row j+24, columns 10975 to 10832 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the base block 1.

The 38 FlexO-x-SC overhead bits in FBG row j+48, columns 10657 to 10694 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the base block 2.

The 38 FlexO-x-SC overhead bits in FBG row j+72, columns 10519 to 10556 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the base block 3.

The 38 FlexO-x-SC overhead bits in FBG row j+96, columns 10381 to 10418 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the base block 4.

The 38 FlexO-x-SC overhead bits in FBG row j+119, columns 10933 to 10970 are mapped to the payload information bits in row 7, columns 30520 to 30557 of the base block 5.

The last 34 bits in row 7, columns 30558 to 30591 of the payload information area of base blocks 1 to 5 are always assumed with an all-zero value for the FEC parity calculations and not transported by FlexO-x-SC.

The total number of FEC parity bits in the 238 FlexO frame row sequence is:

$$119 \times 690 = 82{,}110 \text{ bit} = 5 \times 16{,}422 = 5 \times [38 + 16{,}384] = 5 \times [38 + (512 \times 32)]$$

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of base block 1 are mapped to the 16,384 parity bits of B_{i-2} in FBG row j+1, column 10281 to row j+24, column 10794 (see Figure C.1).

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of base block 2 are mapped to the 16,384 parity bits of B_{i-1} in FBG row j+24, column 10833 to row j+48, column 10656.

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of base block 3 are mapped to the 16,384 parity bits of B_i in FBG row j+48, column 10695 to row j+72, column 10518.

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of base block 4 are mapped to the 16,384 parity bits of B_{i+1} in FBG row j+72, column 10557 to row j+96, column 10380.

The 16,384 parity bits in row 0, column 30592 to row 7, column 32639 of base block 5 are mapped to the 16,384 parity bits of B_{i+2} in FBG row j+96 column 10419 to row j+119, column 10932.

The bit order of the information bits in columns 1 to 10280 of the FlexO-x-SC frame and the bit order of information bits of the 8×30592 bit information base blocks is the same, and is from left-to-right and top-to-bottom.

The bit order of the parity bits in columns 10281 to 10970 of the FlexO-x-SC frame and the bit order of parity bits of the 8×2048 bit parity base blocks is the same, and is from left-to-right and top-to-bottom.

C.2 200G|400G FlexO-x-SC transmitter and receiver SC FEC processing

Figure C.2 presents a 200G|400G FlexO-x-SC (x = 2,4) specific version of Figure A.1 [ITU-T G.709.2]. In Figure C.2, the "Input frame without FEC parity", "Output frame with SC FEC parity" and "Output frame without FEC parity" blocks within Figure A.1 [ITU-T G.709.2] are replaced by a "200G|400G FlexO-x (x=2,4)" block that contains 244,664 information bits, a "200G|400G FlexO-x-SC (x=2,4)" block that contains 244,664 information, 16384 parity and 38 overhead bits and a "200G|400G FlexO-x" block that contains 244,664 information bits, respectively. Furthermore, it is illustrated that the first 30592 columns of a base block in Figure C.2 contain "32 RES, 6 MBAS and 34 PAD" bits instead of "72 PAD" bits.

A vector of 244,664 information bits from a FBG is distributed and mapped into the first 30592 columns of a base block together with 38 overhead (32 RES + 6 MBAS) bits and 34 PAD bits. Then the staircase FEC specific transmit side processing is performed and 16,384 parity bits become available in the last 2048 columns of a base block. FlexO-x-SC information and overhead (32 RES + 6 MBAS) bits plus computed parity bits are then mapped into the 200G|400G FlexO-x-SC frame format.

At the receive side the information, overhead and parity bits in the 200G|400G FlexO-x-SC signal are mapped into information and parity areas of base blocks. Then the staircase specific receive side processing is performed and decoded information bits are stored in the first 30592 columns of a base block. The information bits are then mapped to the 200G|400G FlexO-x frame format.

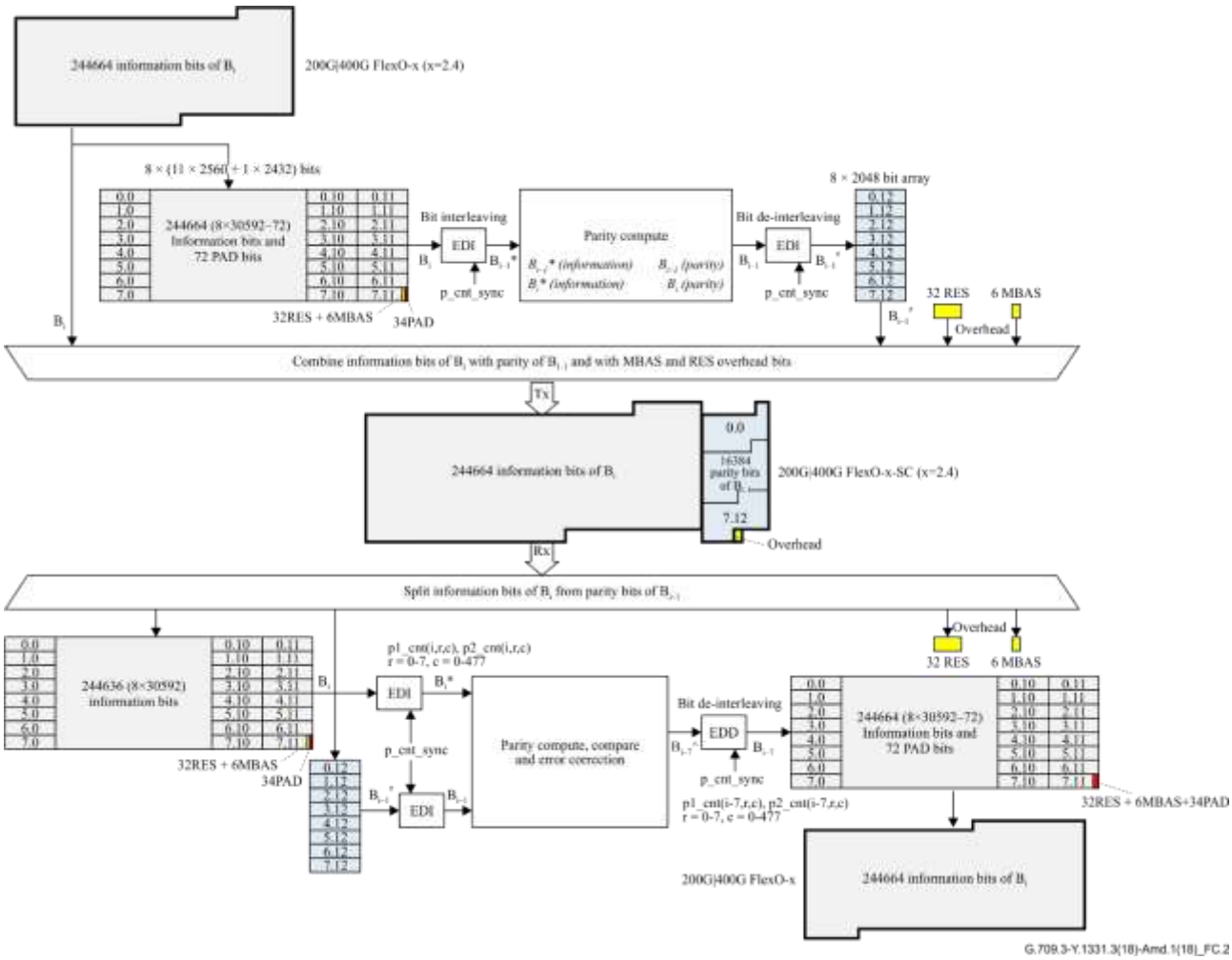


Figure C.2 – 200G|400G FlexO-x-SC transmitter and receiver SC FEC processing

Annex D

Forward error correction using 10976 × 128 Hamming soft decision codes

(This annex forms an integral part of this Recommendation.)

D.1 Forward error correction code

The forward error correction for the FlexO-x-DSH uses a systematic (128,119) double-extended Hamming code.

For FEC processing, the FBG and 714 bit PAD block structure is separated into 119-bit sub-blocks as shown in Figures 15-13 and 15-14. The 9 FEC parity check bits are calculated over the information bits 1 to 119 of each sub-block and appended to the 119 bit blocks as bits 120 to 128 as shown in Figure D.1 and Figure 15-3.

Figure D.1 – (119 + 9)-bit Hamming SD FEC blocks

The systematic double-extended Hamming code is most naturally defined in terms of its parity-check matrix. Consider the function g which maps an integer $i, 0 \le i \le 127$, to the column vector

$$g(i) = \begin{bmatrix} s_{0,i} \\ s_{1,i} \\ \vdots \\ s_{6,i} \\ s_{7,i} \\ 1 \end{bmatrix},$$

where $i = 64s_{6,i} + 32s_{5,i} + \cdots + 2s_{1,i} + s_{0,i}$, and

$$s_{7,i} = (s_{0,i} \wedge s_{2,i}) \vee (\overline{S_{0,i}} \wedge \overline{S_{1,i}} \wedge \overline{S_{2,i}}) \vee (s_{0,i} \wedge s_{1,i} \wedge \overline{S_{2,i}}).$$

The parity-check matrix is then a 9×128 binary matrix:

$$H = [g(0){:}\, g(62), g(64){:}\, g(94), g(96){:}\, g(110), g(112){:}\, g(118), g(120), g(122), g(124),$$
$$g(63), g(95), g(111), g(119), g(121), g(123), g(125){:}\, g(127)]$$

where g(a):g(b) represents [g(a),g(a+1),g(a+2),…,g(b)].

To obtain the encoder matrix G, we calculate

$$P = B[g(0){:}\, g(62), g(64){:}\, g(94), g(96){:}\, g(110), g(112){:}\, g(118), g(120), g(122), g(124)],$$

where:

$$\text{B=}[\, g(63), g(95), g(111), g(119), g(121), g(123), g(125){:}\, g(127)]^{-1}.$$

Finally, the generator matrix of the Hamming code is

$$G = [I; P^T],$$

and a 119-bit message $b = [b_0, b_1, \ldots, b_{118}]$ is encoded to the 128-bit code word

$$c = [c_0, c_1, \ldots, c_{127}] = bG.$$

Annex E

Forward error correction using extended BCH(256,239) soft decision code

(This annex forms an integral part of this Recommendation.)

E.1 Forward error correction code

Forward error correction for the FlexO-x-DO uses a spatially coupled TPC-like code with BCH(256,239) constituent code.

Consider the FEC processing to generate the matrix Vi{t} at time "t", where the matrix Ui and the previous instance of matrix Vi{t-1} are already generated. The encoding of each extended *BCH*(256,239) code word Wi(t,P,p), with P $\in$ {0..21×Z-1} and p $\in$ {0..31} is done sequentially in order of increasing rows, as follows.

First, the leftmost 128 bits Wi(t,P,p,k, k = 0..127) of the code word Wi(t,P<10,p) are determined based on bits in matrix Vi{t-1} and Vi{t}, while the leftmost 128 bits of the code word Wi(t,P≥10,p) are determined based on the bits in matrix Vi{t} as specified in equation 16-2. Next, the 111 bits Wi(t,P,p,k, k = 128..238) of the code word Wi(t,P,p) are determined based on the bits in matrix Ui as specified in equation 16-3. Then the 17 FEC parity check bits Wi(t,P,p,k, k = 239..255) are computed over the information bits 0 to 238 of each W(t,P,p) such that Wi(t,P,p)H = 0 using a textbook encoding, where H is the parity check matrix of the extended *BCH*(256, 239) code that has a minimum Hamming distance 6. Note that if *x* is a vector satisfying *x*H = 0, then,

1. *x* has an even parity, and
2. if the first 255 bits of *x* are seen as the coefficients of a binary polynomial of degree 254 (with bit 0 of *x* being the coefficient of power 254), then this polynomial is divisible by the binary polynomial $y^{16} + y^{14} + y^{13} + y^{11} + y^{10} + y^9 + y^8 + y^6 + y^5 + y + 1$.

The computed 17 FEC parity check bits are appended to the 239 bit blocks as bits 239 to 255 as shown in Figure E.1 and Figure 16-11 resulting in the extended matrix Ui (eUi).

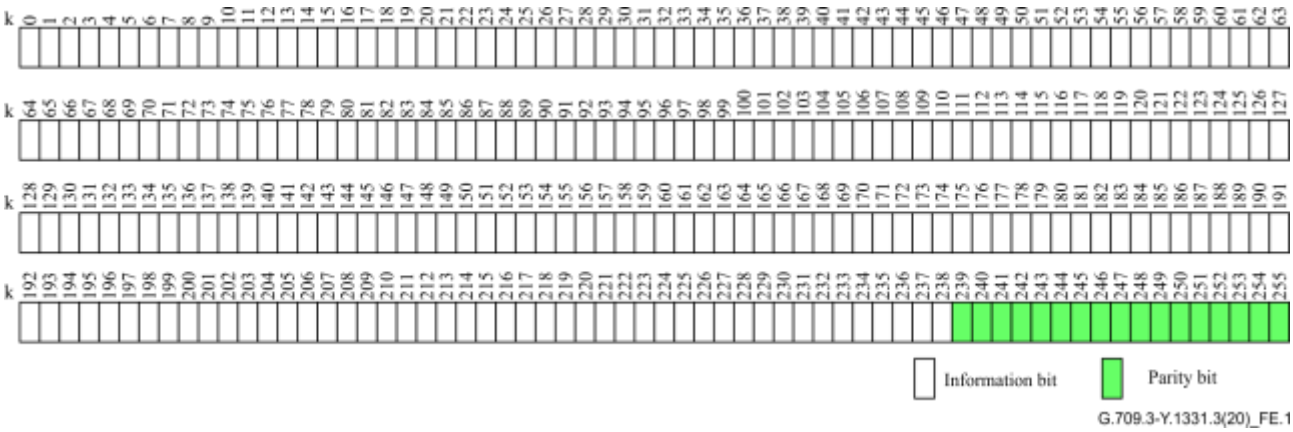


Figure E.1 – Wi(P,p) structure

A code word in OFEC is a semi-infinite set of bits organized in a matrix with semi-infinite number of rows and 128 columns.

It has the property that each bit is part of two "constituent code words," in which each constituent code word is a binary vector *x* of length 256 satisfying the constraint *x*H = 0.

The fraction of bits that are parity bits is 17/128, the rate of the code is 111/128 = 0.867, and the overhead is 17/111 = 15.3%.

OFEC decoding

Any of the iterative algorithms designed for turbo decoding of Product Codes can easily be adapted to decode OFEC code words.

For use with iterative decoding, observe that the bits in block row R will all have been decoded as front bits in later constituent code words after 2 × (8 + 2 + 1) = 22 rows of blocks have been decoded. Specifically, in Figure 16-11, bits in row R = 2P-20 will all have been decoded as front bits by the time block row R = 2P+1 has been decoded. It then makes sense to decode the constituent codes in block row R = 2P – 20 again.

Annex F

Multiplexing OTUCn$_i$ signals into payload of n FlexO instances

(This annex forms an integral part of this Recommendation.)

A set of n FlexO instances may carry multiple OTUCn$_i$ (i = 1..N) signals via GMP. In the most general case, the n = n_1 + n_2 + .. + n_N OTUC instances of the OTUCn$_i$ (i = 1..N) are mapped to a FlexO-x-<fec>-m group of m FlexO-x-<fec> interfaces, each with a FlexO-x-<fec> interface bandwidth of $\lceil n/m \rceil$ *100G. In the default case, the n = n_1 + n_2 + .. + n_N OTUC instances of the OTUCn$_i$ (i = 1..N) are mapped to a single FlexO-x-<fec> signal, with x ≥ n.

F.1 Distributing OTUCn$_i$ and combining OTUC instances

An OTUCn frame structure is specified in clause 11.3 [ITU-T G.709] and contains n synchronous instances of OTUC frame structures. As shown in Figure F.1, the OTUCn$_i$ multiplexing consists of splitting each of the OTUCn$_i$ frames into n_i × OTUC instances and then the mapping of each OTUC instance in a FlexO instance under control of a GMP process. The n_i GMP processes associated with one OTUCn$_i$ may be locked together at the OTUCn$_i$ to FlexO mapper, or could be operated independently. At the demapper, each GMP process is operated independently.

Similarly, the OTUCn$_i$ demultiplexing consists of extracting each OTUC instance from its FlexO instance under control of a GMP process and combining the n_i × OTUC instances of each OTUCn$_i$ into the OTUCn$_i$. Alignment and deskewing are performed on the OTUC instances of each OTUCn$_i$.

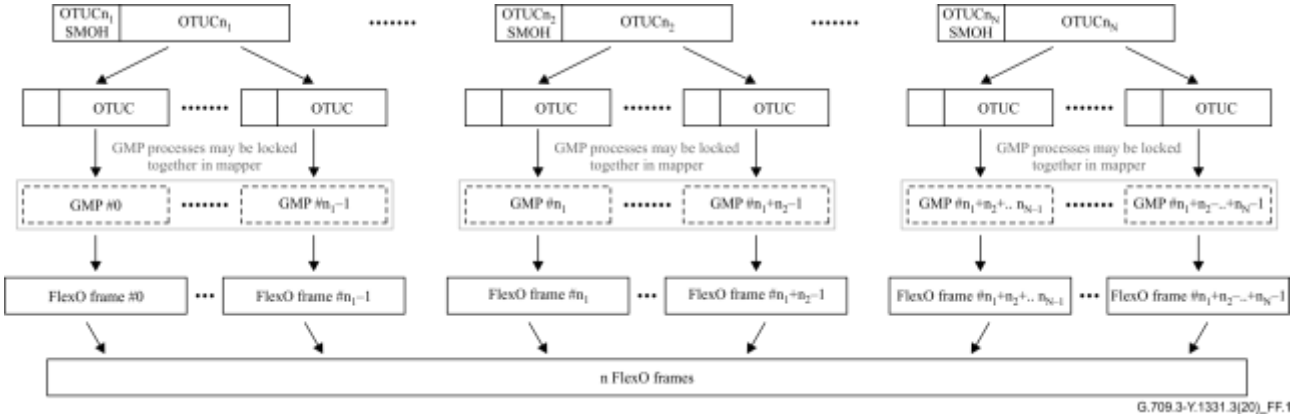


Figure F.1 – Multiple OTUCn$_i$ distributed onto n * FlexO frame instances example

F.2 FlexO frame and 4-frame multi-frame payload structure

The FlexO frame payload area is divided in 256-bit payload blocks (see Figure F.2). The 256-bit payload blocks are aligned to the start of a FlexO payload area (following AM, EOH and BOH). The FlexO frame payload consists of 2565 256-bit blocks.

NOTE – There is no fixed stuff in frames 1 to 7 of the FlexO 8-frame multi-frame in this mapping.

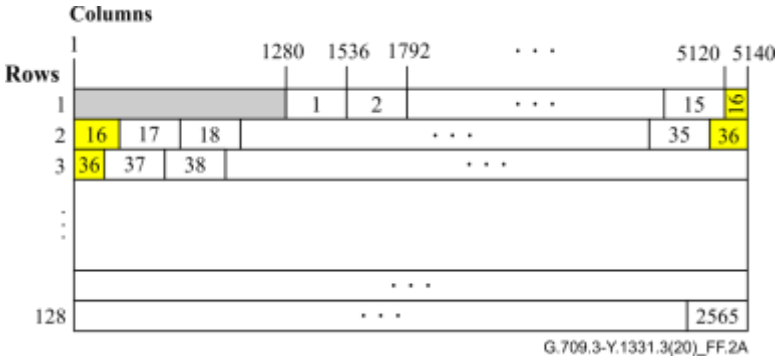


Figure F.2A – FlexO frame payload structure with 256-bit payload blocks

Figure F.2B – FlexO frame payload structure in 128-bit block format with 256-bit GMP blocks

The FlexO payload 4-frame multi-frame structure with 256-bit payload blocks in the payload area for the mapping of an OTUC client signal is illustrated in Figures F.3A and F.3B. The payload area consists of 10,260 256-bit blocks.

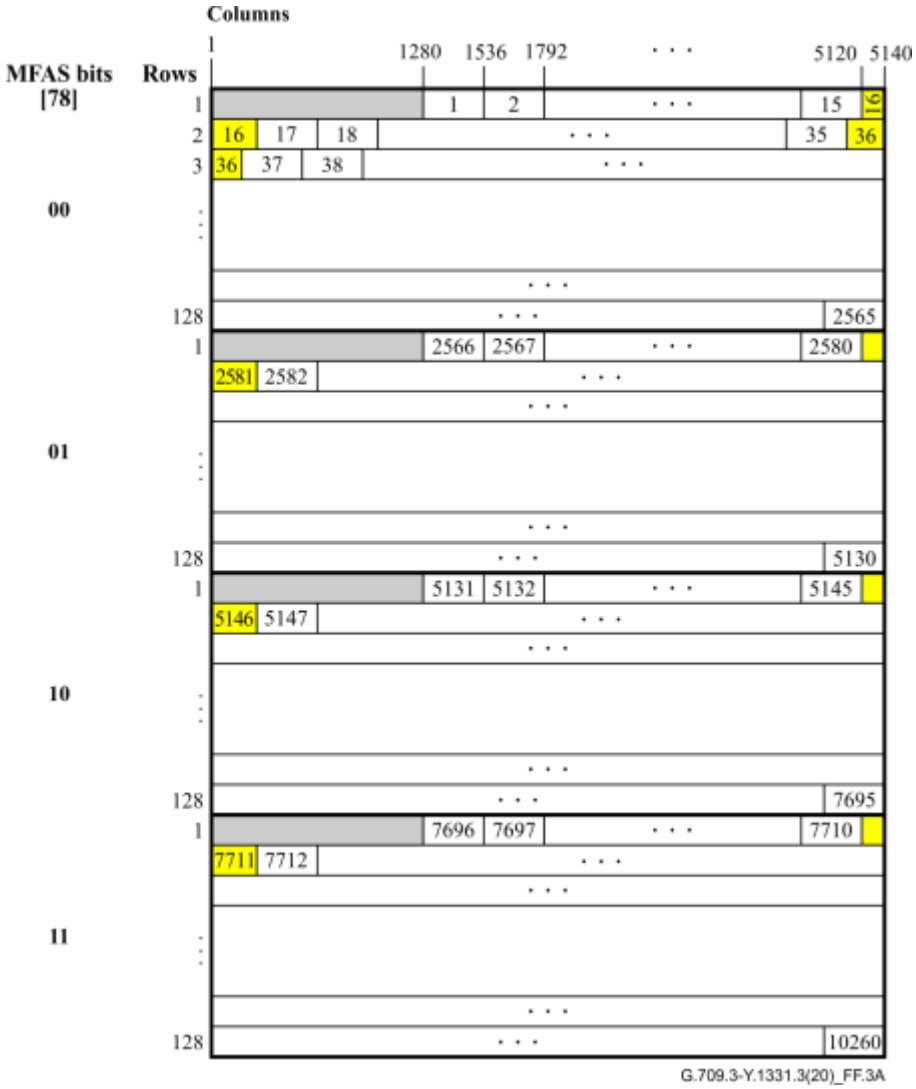


Figure F.3A – FlexO payload 4-frame multi-frame structure with 256-bit payload blocks for the GMP mapping of an OTUC client signal

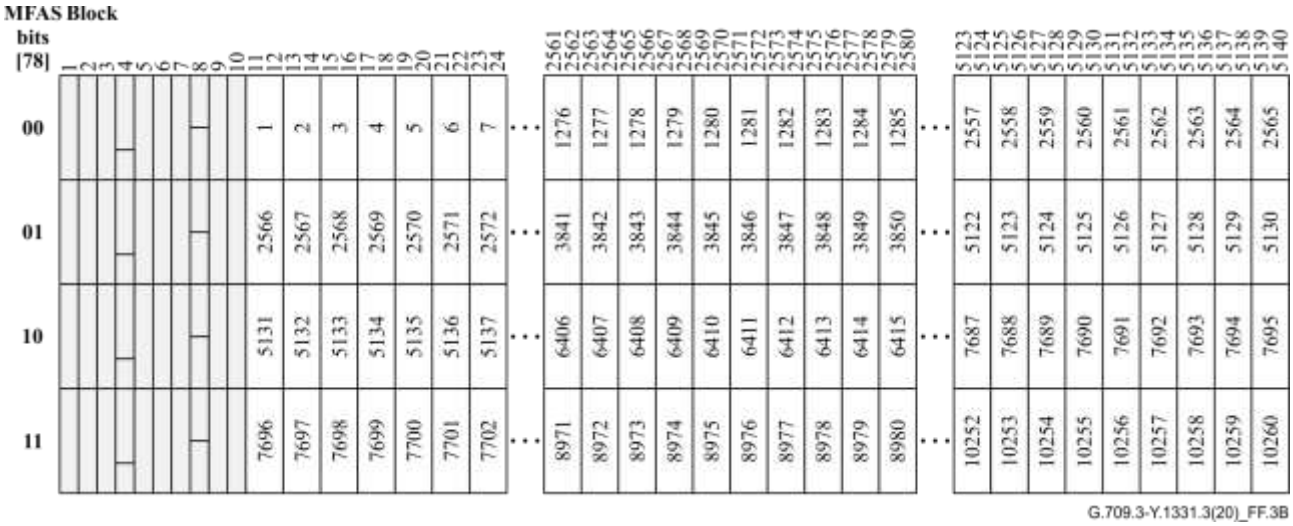


Figure F.3B – FlexO payload 4-frame multi-frame structure in 128-bit block format with 256-bit payload blocks for the GMP mapping of an OTUC client signal

F.3 FlexO client mapping specific overhead

For OTUCn multiplexing, the mapping specific overhead consists of a multiplex structure identifier (MSI) and justification control (JC1-JC6) overhead. The FlexO MSI and JC1-JC6 overhead locations are illustrated in Figure F.4 and are present in each FlexO instance of the group of $n = n_1 + n_2 + .. + n_N$ FlexO instances.

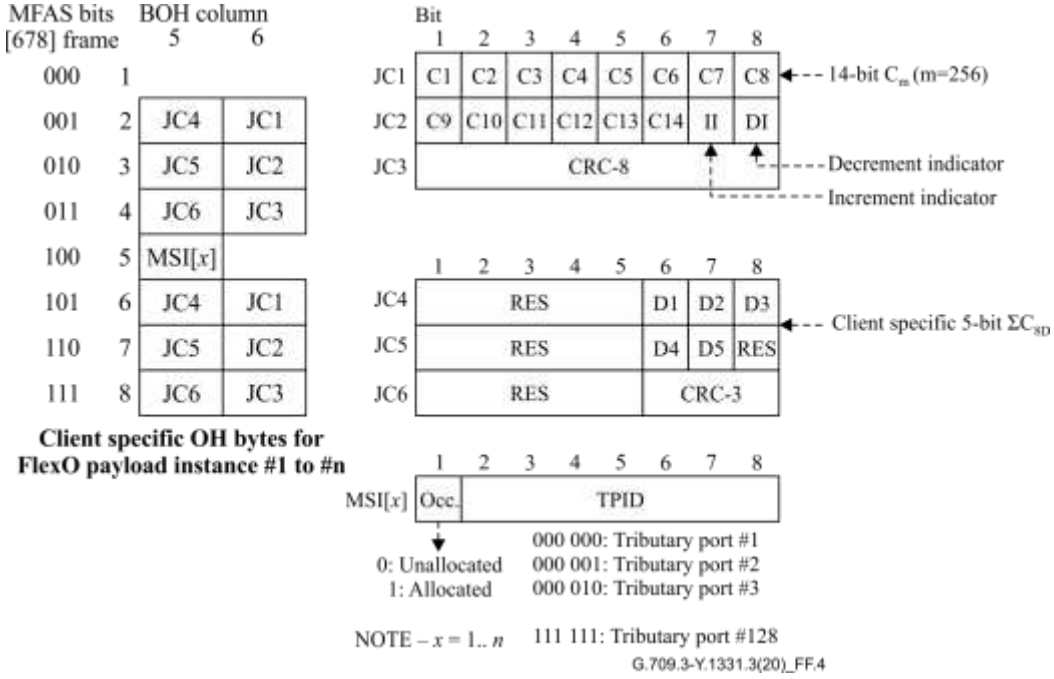


Figure F.4 – FlexO multiplex and justification overhead

F.3.1 FlexO multiplex structure identifier (MSI)

The FlexO multiplex structure identifier (MSI) overhead, which encodes the OTUC multiplex structure in the FlexO-n payload is located in frame 5, in overhead byte 5 in all n FlexO frames, as illustrated in Figure F.4. The MSI indicates the OTUC content of each FlexO instance payload. One byte is used for each FlexO instance.

– The FlexO occupation bit 1 indicates if the FlexO instance payload is allocated or unallocated.

– The tributary port identifier in bits 2 to 8 indicates the tributary port number of the $OTUCn_i$ of which an OTUC instance is being transported in this FlexO instance; for the case of a $OTUCn_i$ carried in two or more FlexO instances a flexible assignment of tributary port to FlexO instance is possible. $OTUCn_i$ tributary ports are numbered 1 to n. The value is set to all-0s when the occupation bit has the value 0 (FlexO instance is unallocated).

F.3.2 FlexO justification overhead

The FlexO justification control overhead (JOH), which carry GMP justification control overhead, is located in frames 2, 3, 4, 6, 7 and 8 in overhead bytes 5 and 6 in all n FlexO frames, as illustrated in Figure F.4. It consists of two times two groups of three bytes of justification control: JC1, JC2, JC3 and JC4, JC5, JC6.

The JC1, JC2 and JC3 bytes consist of a 14-bit $C_m(t)$ field (bits C1 (MSB), C2, .., C14 (LSB)), a 1-bit increment indicator (II) field, a 1-bit decrement indicator (DI) field and a 8-bit CRC-8 field which contains an error check code over the JC1, JC2 and JC3 bytes.

The JC4, JC5 and JC6 bytes consist of a 5-bit ΣC_{nD} field (bits D1, D2, .., D5), a 3-bit CRC-3 field which contains an error check code over bits 6 to 8 in the JC4, JC5 and JC6 fields and sixteen bits reserved for future international standardization (RES).

The value of 'm' in C_m is 256.

The value of 'n' represents the timing granularity of the GMP C_n parameter, which is also present in ΣC_{nD}. The value of n is 8.

The value of C_m controls the distribution of groups of two 128-bit OTUC data blocks into 256-bit GMP blocks in the FlexO payload. Refer to clause F.4 and Annex D of [ITU-T G.709] for further specification of this process.

The value of ΣC_{nD} provides additional 'n'-bit timing information, which is necessary to control the jitter and wander performance experienced by the OTUC signal.

The value of C_n (i.e., number of client n-bit data entities per FlexO payload 4-frame multi-frame) is computed as follows: $C_n(t) = m \times C_m(t) + (\Sigma C_{nD}(t) - \Sigma C_{nD}(t–1))$. Note that the value C_{nD} is effectively an indication of the amount of data in the mapper's virtual queue that it could not send during that multi-frame due to it being less than a 256-bit word. For the case where the value of ΣC_{nD} in a multi-frame 't' is corrupted, it is possible to recover from such error in the next multi-frame 't+1'.

The CRC-8 located in JC3 is calculated over the JC1 and JC2 bits. The CRC-8 uses the $g(x) = x^8 + x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC1 and JC2 octets are taken in network octet order, most significant bit first, to form a 16-bit pattern representing the coefficients of a polynomial *M(x)* of degree 15.
2) *M(x)* is multiplied by x^8 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 7 or less.
3) The coefficients of *R(x)* are considered to be an 8-bit sequence, where x^7 is the most significant bit.
4) This 8-bit sequence is the CRC-8 where the first bit of the CRC-8 to be transmitted is the coefficient of x^7 and the last bit transmitted is the coefficient of x^0.

The de-mapper process performs steps 1-3 in the same manner as the mapper process, except that here, the *M(x)* polynomial of step 1 includes the CRC bits of JC3, resulting in *M(x)* having degree 23. In the absence of bit errors, the remainder shall be 0000 0000.

NOTE 1 – Refer to Appendix VI of [ITU-T G.709] for a parallel logic implementation of CRC-8.

The CRC-3 is calculated over bits 6-8 in JC4 and JC5. The CRC-3 uses the $g(x) = x^3 + x^2 + 1$ generator polynomial, and is calculated as follows:

1) The JC4 bits 6-8 and JC5 bits 6-8 are taken in network transmission order, most significant bit first, to form a 6-bit pattern representing the coefficients of a polynomial *M(x)* of degree 5.
2) *M(x)* is multiplied by x^3 and divided (modulo 2) by *G(x)*, producing a remainder *R(x)* of degree 2 or less.
3) The coefficients of *R(x)* are considered to be a 3-bit sequence, where x^2 is the most significant bit.
4) This 3-bit sequence is the CRC-3 where the first bit of the CRC-3 to be transmitted is the coefficient of x^2 and the last bit transmitted is the coefficient of x^0.

The demapper process performs steps 1-3 in the same manner as the mapper process. In the absence of bit errors, the remainder shall be 000.

NOTE 2 – Refer to clause 6.2.8 of [ITU-T G.7044] for a parallel logic implementation of CRC-3. Replace RCOH1 and RCOH2 bits 1-3 by JC4 and JC5 bits 6-8.

F.4 Mapping of OTUCn$_i$ into n$_i$ FlexO frames

An OTUC instance of an OTUCn$_i$ is associated to a FlexO frame. The 256 successive bits (32 bytes) of an OTUC instance are mapped into one 256-bit GMP block of a FlexO 4-frame multi-frame payload area using a generic mapping procedure (GMP) data/stuff control mechanism as specified in Annex D of [ITU-T G.709]. Each 256-bit GMP block may either carry one 256-bit OTUC block, or carry one 256-bit stuff block. The value of the 256-bit stuff blocks is set to all-0s. The 256-bit block of OTUC is aligned to the OTUC frame structure.

The 256-bit blocks in each FlexO payload area are numbered from 1 to 10260. The FlexO instance payload block numbering for GMP 256-bit blocks is illustrated in Figures F.3A and F.3B. In frame 1 of the FlexO 4-frame multi-frame the first 256-bit block will be labelled 1, the next 256-bit block will be labelled 2, etc.

The values of m, $C_{m,min}$, $C_{m,max}$, n, $C_{n,min}$ and $C_{n,max}$ for OTUCn$_i$ client into FlexO-n$_i$ Payload area are as follows:

$$m = see\ Table\ F.1 \tag{F-1}$$

$$c_{m,nom} = \left(\frac{OTUC\_nom\_bit\_rate \times Number\_of\_GMP\_blocks\_in\_FlexO-payload}{FlexO-payload\_nom\_bit\_rate}\right) \tag{F-2}$$

$$c_{m,min} = c_{m,nom} \times \left(\frac{1 - OTUC\_bit\_rate\_tolerance}{1 + FlexO-payload\_bit\_rate\_tolerance}\right) \tag{F-3}$$

$$c_{m,max} = c_{m,nom} \times \left(\frac{1 + OTUC\_bit\_rate\_tolerance}{1 - FlexO\_payload\_bit\_rate\_tolerance}\right) \tag{F-4}$$

$$C_{m,min} = \lfloor c_{m,min} \rfloor \tag{F-5}$$

$$C_{m,max} = \lceil c_{m,max} \rceil \tag{F-6}$$

$$n = see\ Table\ F.1 \tag{F-7}$$

$$c_{n,nom} = \left(\frac{OTUC\_nom\_bit\_rate \times Number\_of\_GMP\_blocks\_in\_FlexO-payload}{FlexO-payload\_nom\_bit\_rate}\right) \tag{F-8}$$

$$c_{n,min} = c_{n,nom} \times \left(\frac{1 - OTUC\_bit\_rate\_tolerance}{1 + FlexO-payload\_bit\_rate\_tolerance}\right) \tag{F-9}$$

$$c_{n,max} = c_{n,nom} \times \left(\frac{1 + OTUC\_bit\_rate\_tolerance}{1 - FlexO-payload\_bit\_rate\_tolerance}\right) \tag{F-10}$$

$$C_{n,min} = \lfloor c_{n,min} \rfloor \tag{F-11}$$

$$C_{n,max} = \lceil c_{n,max} \rceil \tag{F-12}$$

$C_{m,min}$, $C_{n,min}$, $C_{m,max}$ and $C_{n,max}$ values represent the boundaries of OTUC/FlexO payload ppm offset combinations (i.e., min. client/max. FlexO-n$_i$ payload and max. OTUC/min. FlexO payload). In steady state, given instances of OTUC/FlexO payload offset combinations should not result in generated C_m and C_n values throughout this range but rather should be within as small a range as possible.

Under transient ppm offset conditions (e.g., replacement signal to normal signal), it is possible that C_n and C_m values outside the range $C_{n,min}$ to $C_{n,max}$ and $C_{m,min}$ to $C_{m,max}$ may be generated and a GMP de-mapper should be tolerant of such occurrences.

Table F.2 gives the GMP parameter values for OTUC mapping into FlexO.

Table F.1 – m, n and C_{nD} for $OTUCn_i$ clients into $n_i \times$ FlexO payload

Client signal	Nominal bit rate (kbit/s)	Bit-rate tolerance (ppm)	m	n	C_{nD}
$OTUCn_i$	$n_i \times 239/226 \times 99\,532\,800$ kbit/s	±20	$n_i \times 256$	8	Yes
NOTE – The nominal $OTUCn_i$ bit rate is approximately: $n_i \times 105\,258\,138.053$ kbit/s.					

Table F.2 – GMP parameter values for OTUC mapping into FlexO

Ref	GMP Parameter	Formula	Value	Units
F_{client}	nominal OTUC signal bit rate	99.5328 Gbit/s × 239/226	105,258,138,053.097	bit/s
Δf_{client}	client signal bit rate tolerance		20	ppm
f_{server}	FlexO server nominal bit rate f	99.5328 Gbit/s × 491384/462961	105,643,510,782.118	bit/s
Δf_{server}	server bit rate tolerance		20	ppm
T_{server}	period of the server multi-frame,	B_{server} / f_{server}	24.911	µs
B_{server}	number of bits per server multi-frame		2,631,680	bits
O_{server}	number of overhead bits per server multi-frame		5,120	bits
P_{server}	maximum number of bits in the server payload area	$B_{server} - O_{server}$	2,626,660	bits
$f_{p,server}$	nominal server payload bit rate	$f_{server} \times P_{server} / B_{server}$ = 99.5328 Gbit/s × 491384/462961 × 513/514	105,437,978,659.973	bits/s
m	GMP data/stuff granularity	m bit data entity	256	bits
M	m and n ratio	m / n	32	
$P_{m,server}$	maximum number of (m bits) data entities in the server payload area	P_{server} / m	10260	256b blocks
c_m	number of client m-bit data entities per server multi-frame			
$c_{m,nom}$	c_m value at nominal client and server bit rates	$(f_{client} / f_{p,server}) \times P_{m,server}$	10,242.500	
$c_{m,min}$	c_m value at minimum client and maximum server bit rates	$c_{m,nom} \times (1 - \Delta f_{client}) / (1 + \Delta f_{server})$	10,242.092	
$c_{m,max}$	c_m value at maximum client and minimum server bit rates	$c_{m,nom} \times (1 + \Delta f_{client}) / (1 - \Delta f_{server})$	10,242.910	
$C_{m,min}$	integer value of $c_{m,min}$	$\lfloor c_{m,min} \rfloor$	10,242	
$C_{m,max}$	rounded up value of $c_{m,max}$	$\lceil c_{m,max} \rceil$	10,243	
n	GMP justification accuracy,		8	bits

Table F.2 – GMP parameter values for OTUC mapping into FlexO

Ref	GMP Parameter	Formula	Value	Units
	n bit data entity			
$P_{n,server}$	maximum number of (n bits) data entities in the server payload area	P_{server} / n	328,320.000	8b blocks
c_n	number of client n-bit data entities per server multi-frame			
$c_{n,nom}$	C_n value at nominal client and server bit rates	$(f_{client} / f_{p,server}) \times P_{n,server}$	327,760.000	
$c_{n,min}$	C_n value at minimum client and maximum server bit rates	$C_{n,nom} \times (1 - \Delta f_{client}) / (1 + \Delta f_{server})$	327,746.890	
$c_{n,max}$	C_n value at maximum client and minimum server bit rates	$C_{n,nom} \times (1 + \Delta f_{client}) / (1 - \Delta f_{server})$	327,773.111	
c_{nD}	remainder of c_n and C_m	$c_n - (m/n \times C_m)$		
C_{nD}	integer value of c_{nD}	$\lfloor c_{nD} \rfloor$		
$\sum C_{nD}$	accumulated value of C_{nD}	0 to m/n – 1	31	

Annex G

FlexO-x-D<fec> TS, PS and MFAS overhead values

(This annex forms an integral part of this Recommendation.)

This annex specifies the bit values of the TS, PS and MFAS overhead within the FlexO-x-DSH and FlexO-x-DO frame structures. ~~This version of the annex contains place holders for the bit values of the TS, PS and MFAS overhead within the FlexO-x-DO frame structure.~~

G.1 FlexO-x-DSH and FlexO-x-DO TS, PS and MFAS overhead values

G.1.1 Training sequence (TS)

The values of the 11 TS overhead blocks are specified in Table G.1.

For the case of FlexO-x-DSH, these values are associated with bits 1,2,3,4 and 5,6,7,8 (Z = 8) and bits 1,2 and 3,4 (Z = 4). For the case of FlexO-x-DO, these values are associated with bits 1,3,5,7 and 2,4,6,8 (Z = 8) and bits 1,3 and 2,4 (Z = 4).

Table G.1 – FlexO-x-DSH and FlexO-x-DO training sequence values

Column #	TS value (Z=8)	TS value (Z=4)
	(DSH) 1234 5678 (DSH) (DO) 1357 2468 (DO)	(DSH) 12 34 (DSH) (DO) 13 24 (DO)
1	0010 0000	01 00
2	1010 0000	11 00
3	0010 1000	01 10
4	1010 0010	11 01
5	0000 0010	00 01
6	1010 1010	11 11
7	0000 0000	00 00
8	0000 0010	00 01
9	1010 1000	11 10
10	1000 1010	10 11
11	1000 1000	10 10

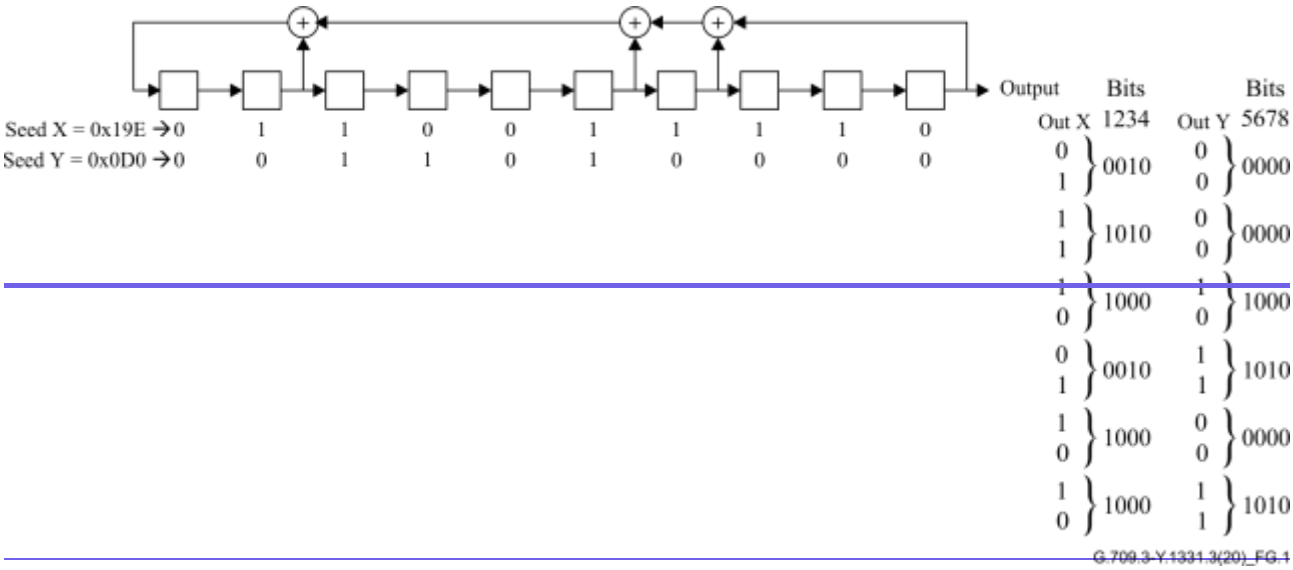
G.1.2 Pilot sequence (PS)

G.1.2.1 8-bit block (Z=8)

The values of the 116 8-bit PS overhead blocks are specified in Table G.2. For the case of FlexO-x-DSH, these values are associated with bits 1,2,3,4 and 5,6,7,8. For the case of FlexO-x-DO, these values are associated with bits 1,3,5,7 and 2,4,6,8.

The values of bits 1,2,3, ~~to~~ 4 (for DSH) and bits 1,3,5,7 (for DO) are derived from a fixed PRBS10 sequence with a seed of X = 414 (0x19E). The values of bits 5,6,7, ~~to~~ 8 (for DSH) and 2,4,6,8 (for DO) are derived from a fixed PRBS10 sequence with a seed of Y = 208 (0x0D0). The seed is reset at the start of every FlexO-x-D<fec> frame. The PRBS10 sequence polynomial is: $x^{10} + x^8 + x^4 + x^3 + 1$.

Every 2 bits from the PRBS10 sequence are encoded as 4 bits as follows: 00 → 0000, 01 → 0010, 10 → 1000 and 11 → 1010. See Figure G.1 for an illustration.



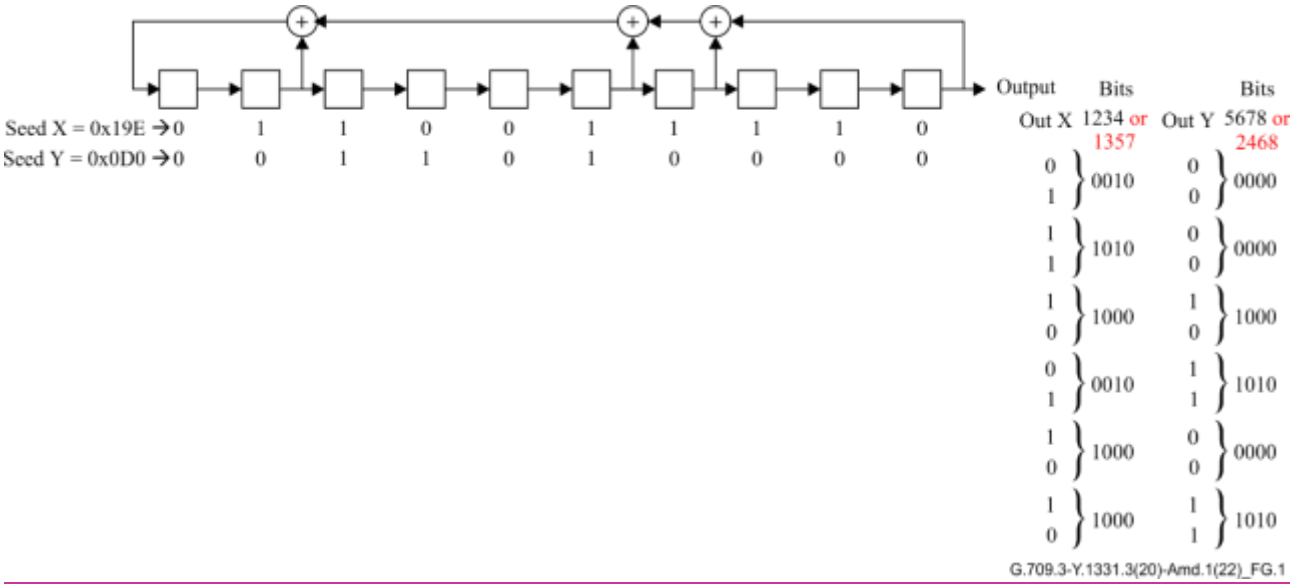


Figure G.1 – FlexO-x-DSH Pilot sequence overhead generator for Z=8

Table G.2 – FlexO-x-DSH and FlexO-x-DO PS overhead values for Z=8

Row #	PS values (DSH) 1234 (DO) 1357	PS values 5678 (DSH) 2468 (DO)	Row #	PS values (DSH) 1234 (DO) 1357	PS values 5678 (DSH) 2468 (DO)	Row #	PS values (DSH) 1234 (DO) 1357	PS values 5678 (DSH) 2468 (DO)
1	0010	0000	40	1000	0000	79	1010	0000
2	1010	0000	41	0000	1000	80	1010	0000
3	1000	1000	42	1000	1000	81	1010	1000
4	0010	1010	43	0010	0000	82	0000	0000
5	1000	0000	44	0010	0000	83	0000	1010
6	1000	1010	45	0000	1010	84	1010	0000
7	0000	0010	46	0010	0010	85	1000	0000
8	1010	0010	47	0000	1010	86	0010	0000

Table G.2 – FlexO-x-DSH and FlexO-x-DO PS overhead values for Z=8

Row #	PS values (DSH) 1234 (DO) 1357	PS values 5678 (DSH) 2468 (DO)	Row #	PS values (DSH) 1234 (DO) 1357	PS values 5678 (DSH) 2468 (DO)	Row #	PS values (DSH) 1234 (DO) 1357	PS values 5678 (DSH) 2468 (DO)
9	0010	0000	48	1010	0010	87	1010	1000
10	1010	1010	49	1010	1000	88	1000	0010
11	1010	1010	50	0010	0010	89	0000	0010
12	0000	0000	51	1000	1010	90	1000	1000
13	1010	1010	52	1000	0010	91	1000	1010
14	1000	1010	53	1000	0010	92	0010	1000
15	1010	1000	54	0000	1010	93	0000	1000
16	1000	1010	55	1000	0010	94	1010	0010
17	1010	1010	56	1010	0010	95	0000	1000
18	1000	0010	57	0010	0000	96	0000	1000
19	0010	0000	58	0000	1000	97	1010	0010
20	0000	1000	59	1000	1000	98	0010	1000
21	1010	1000	60	1010	0010	99	1000	0000
22	0010	1010	61	1000	1010	100	0000	1010
23	0010	0010	62	0000	0000	101	1010	0000
24	1000	1000	63	1000	1010	102	0010	0010
25	0010	1000	64	0010	0010	103	0000	0010
26	0010	1010	65	1000	1000	104	0000	1010
27	0010	0010	66	1010	1010	105	1010	0010
28	0010	1010	67	1000	0000	106	1000	1000
29	0000	1010	68	0010	1000	107	1010	1010
30	1000	1000	69	1000	0010	108	0010	0010
31	0000	0010	70	0010	0010	109	0000	1010
32	1010	0000	71	1010	0010	110	0010	0000
33	0010	1000	72	0000	0000	111	0000	0010
34	0010	0000	73	0000	0010	112	0010	1000
35	0010	0000	74	1000	1010	113	0010	0010
36	1000	1000	75	0010	0000	114	1010	1010
37	1000	1000	76	1000	0000	115	1010	1000
38	0000	0000	77	0010	0000	116	0000	1000
39	0000	1010	78	0000	1010			

G.1.2.2 4-bit block (Z=4)

The values of the 116 4-bit PS overhead blocks are specified in Table G.3. For the case of FlexO-x-DSH, these values are associated with bits 1,2 and 3,4. For the case of FlexO-x-DO these values are associated with bits 1,3 and 2,4.

The values of bits 1, ~~to~~ 2 (for DSH) and bits 1,3 (for DO) are derived from a fixed PRBS10 sequence with a seed of X = 414 (0x19E). The values of bits 3, ~~to~~ 4 (for DSH) and bits 2,4 (for DO) are derived from a fixed PRBS10 sequence with a seed of Y = 208 (0x0D0). The seed is reset at the start of every FlexO-x-D<fec> frame. The PRBS10 sequence polynomial is: $x^{10} + x^8 + x^4 + x^3 + 1$.

See Figure G.2 for an illustration.

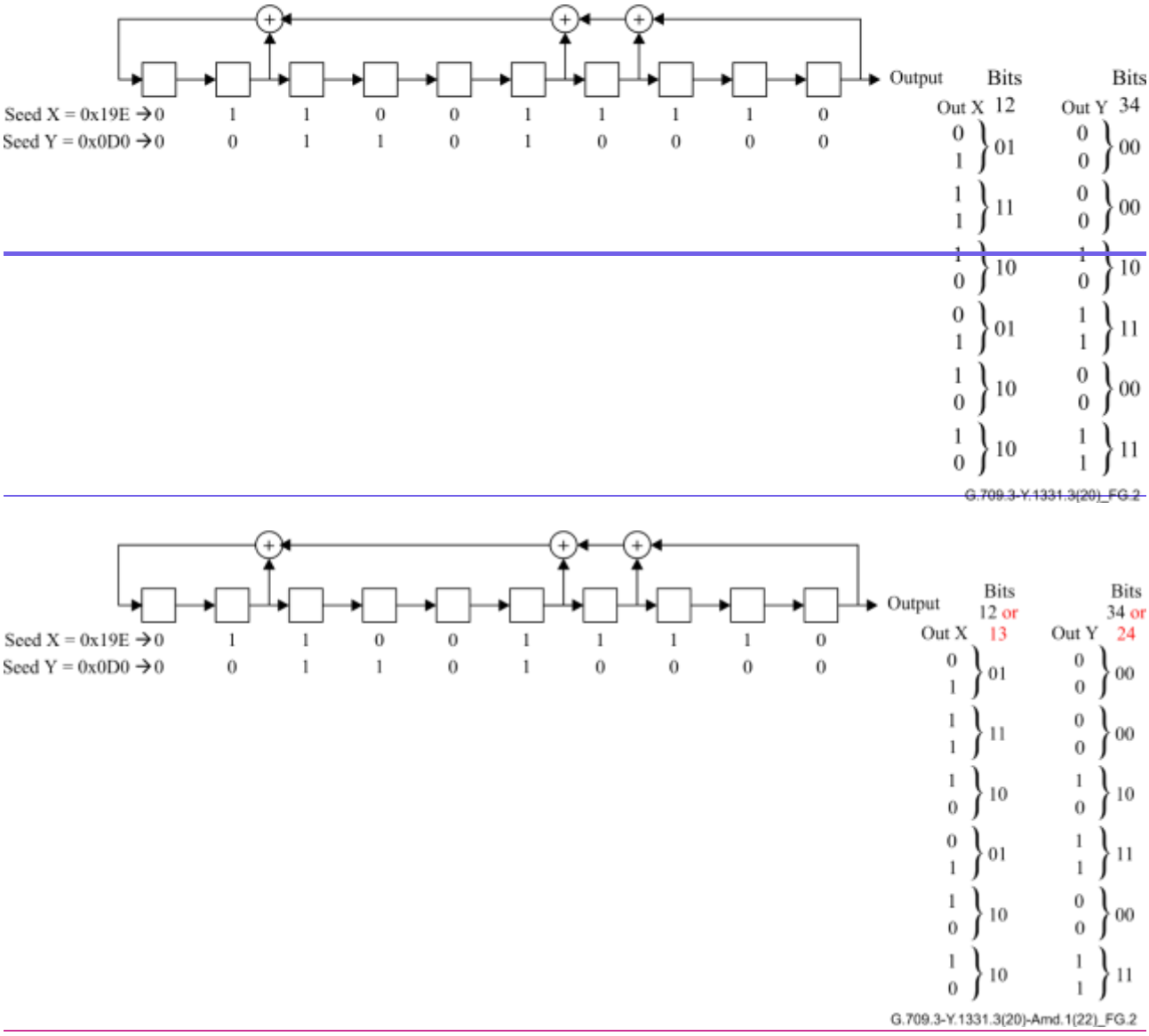


Figure G.2 – FlexO-x-DSH and FlexO-x-DO Pilot sequence overhead generator for Z=4

Table G.3 – FlexO-x-DSH PS overhead values for Z=4

Row #	PS values		Row #	PS values		Row #	PS values	
	(DSH) 12 (DO) 13	34 (DSH) 24 (DO)		(DSH) 12 (DO) 13	34 (DSH) 24 (DO)		(DSH) 12 (DO) 13	34 (DSH) 24 (DO)
1	01	00	40	10	00	79	11	00
2	11	00	41	00	10	80	11	00
3	10	10	42	10	10	81	11	10
4	01	11	43	01	00	82	00	00
5	10	00	44	01	00	83	00	11
6	10	11	45	00	11	84	11	00
7	00	01	46	01	01	85	10	00
8	11	01	47	00	11	86	01	00
9	01	00	48	11	01	87	11	10
10	11	11	49	11	10	88	10	01
11	11	11	50	01	01	89	00	01
12	00	00	51	10	11	90	10	10
13	11	11	52	10	01	91	10	11
14	10	11	53	10	01	92	01	10
15	11	10	54	00	11	93	00	10
16	10	11	55	10	01	94	11	01
17	11	11	56	11	01	95	00	10
18	10	01	57	01	00	96	00	10
19	01	00	58	00	10	97	11	01
20	00	10	59	10	10	98	01	10
21	11	10	60	11	01	99	10	00
22	01	11	61	10	11	100	00	11
23	01	01	62	00	00	101	11	00
24	10	10	63	10	11	102	01	01
25	01	10	64	01	01	103	00	01
26	01	11	65	10	10	104	00	11
27	01	01	66	11	11	105	11	01
28	01	11	67	10	00	106	10	10
29	00	11	68	01	10	107	11	11
30	10	10	69	10	01	108	01	01
31	00	01	70	01	01	109	00	11
32	11	00	71	11	01	110	01	00
33	01	10	72	00	00	111	00	01
34	01	00	73	00	01	112	01	10
35	01	00	74	10	11	113	01	01
36	10	10	75	01	00	114	11	11
37	10	10	76	10	00	115	11	10

Table G.3 – FlexO-x-DSH PS overhead values for Z=4

Row #	PS values (DSH) 12 (DO) 13	PS values 34 (DSH) 24 (DO)	Row #	PS values (DSH) 12 (DO) 13	PS values 34 (DSH) 24 (DO)	Row #	PS values (DSH) 12 (DO) 13	PS values 34 (DSH) 24 (DO)
38	00	00	77	01	00	116	00	10
39	00	11	78	00	11			

G.1.3 Multi-frame alignment signal (MFAS)

The values of the 22 MFAS overhead Z-bit blocks are specified in Table G.4. For the case of FlexO-x-DSH, these values are associated with bits 1,2,3,4 and 5,6,7,8 (Z = 8) and bits 1,2 and 3,4 (Z = 4). For the case of FlexO-x-DO, these values are associated with bits 1,3,5,7 and 2,4,6,8 (Z = 8) and bits 1,3 and 2,4 (Z = 4).

Table G.4 – FlexO-x-DSH MFAS sequence values

(Row,Column) #	MFAS values (Z=8) (DSH) 1234 (DO) 1357	MFAS values (Z=8) 5678 (DSH) 2468 (DO)	MFAS values (Z=4) (DSH) 12 (DO) 13	MFAS values (Z=4) 34 (DSH) 24 (DO)
(1,12)	1000	1010	10	11
(1,13)	1010	0010	11	01
(1,14)	1010	0000	11	00
(1,15)	1010	0010	11	01
(1,16)	1000	1000	10	10
(1,17)	1000	1010	10	11
(1,18)	0000	1000	00	10
(1,19)	1010	1000	11	10
(1,20)	0000	0000	00	00
(1,21)	0010	1000	01	10
(1,22)	0010	1010	01	11
(1,23)	1000	0010	10	01
(1,24)	0000	0010	00	01
(1,25)	0000	1010	00	11
(1,26)	0010	0000	01	00
(1,27)	1010	1010	11	11
(1,28)	0000	0000	00	00
(1,29)	1000	0010	10	01
(1,30)	0010	1000	01	10
(1,31)	1010	0000	11	00
(1,32)	0000	1000	00	10
(2,2)	0010	0010	01	01

~~G.2 FlexO-x-DO TS, PS and MFAS overhead values~~

~~G.2.1 Training sequence (TS)~~

~~For further study.~~

~~G.2.2 Pilot sequence (PS)~~

~~For further study.~~

~~G.2.3 Multi-frame alignment signal (MFAS)~~

~~For further study.~~

Appendix I

Example applications

(This appendix does not form an integral part of this Recommendation.)

FlexO-x-<fec> (<fec> = SC or DSH) group interfaces can be used for a variety of applications.

Example applications for a FlexO-x-SC-m interface group are shown in Figure I.1 and Figure I.2. Such an interface group might represent an OTN handoff between router (R) and transport (T) nodes within one administrative domain, or could be a handoff between OTN equipment of different vendors in one administrative domain.

Optical transport networks are typically subdivided into metro and core where the core interconnects metro networks. Transport services may stay in one metro network or they may extend over different ones. In the latter case they may be passed through the core network.

Network elements in the metro network play different roles such as metro/core gateway, edge towards customer and transit nodes. The customer facing functions lead to some diversity of client interfaces. Network elements from different vendors may be used to serve this broad scope of functions. FlexO-LR interfaces could be used to interconnect network elements of different vendors or the same vendor.

Figure I.1 illustrates an OTN core network with associated metro networks. The figure shows:

– OTN ODU cross connect nodes with electrical switch fabric (labelled EXC) of vendor Z or X interconnected with EXC of vendor X using a FlexO-x-<fec>-m interface group

– packet switching nodes (labelled Router) of vendor Z interconnected with the router of vendor X using a FlexO-x-<fec>-m interface group

– interconnection of the above EXC or router nodes through a metro OTN network

– interconnection of the above EXC or router nodes, establishing a path over which the OCh or OTSiA overhead can be exchanged as specified in [ITU-T G.872], [ITU-T G.709] and [b-ITU-T G.7712] enabling end-to-end optical path monitoring

– interconnection in backbone/core network is also possible if the FlexO-x-<fec> FEC is adequate

Details of the optical path passed by the FlexO-x-<fec> signals are defined in [ITU-T G.698.2].

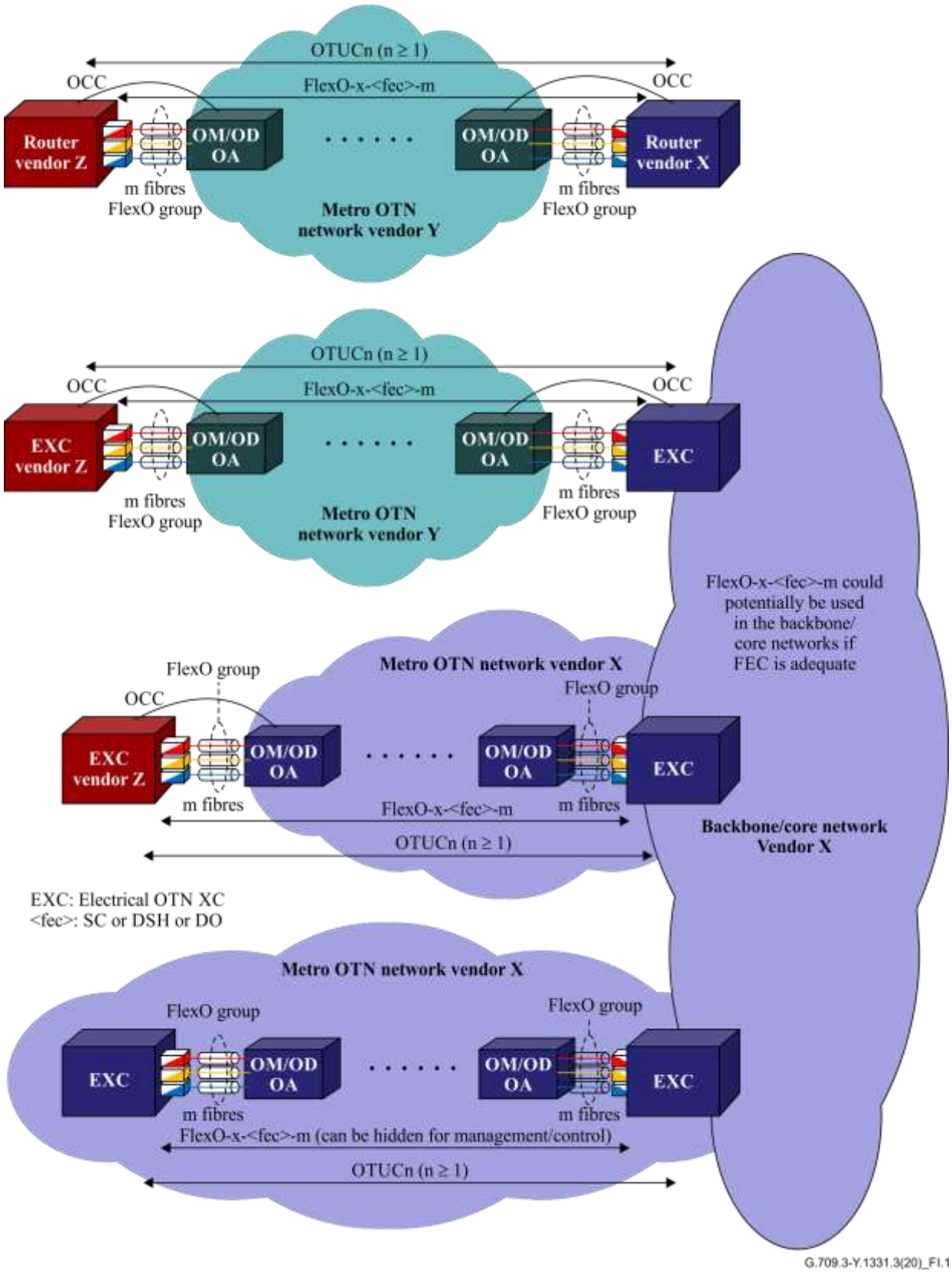


Figure I.1 – Example FlexO-x-<fec>-m deployments in one administrative domain

Figure I.2 illustrates interconnection of the EXC or router nodes through a set of m point-to-point fibres, establishing an inter-domain group interface beyond the distances supported by the FlexO-x-RS-m specified in [ITU-T G.709.1] and [b-ITU-T G.959.1].

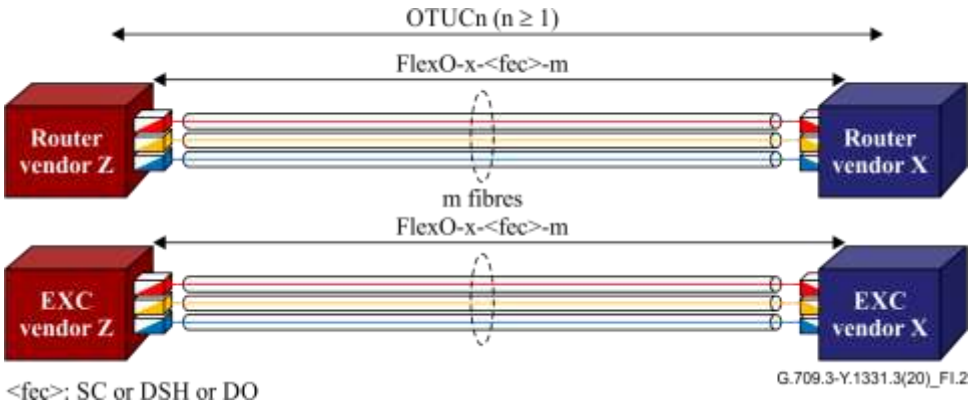


Figure I.2 – Example FlexO-x-<fec>-m deployments establishing an inter-domain group interface beyond the distances supported by the FlexO-x-RS-m

Appendix II

Error correction capability of the (128,119) Hamming soft decision code combined with the 512 × 510 staircase code

(This appendix does not form an integral part of this Recommendation.)

Based on the terms and definitions given in clause 7 of [ITU-T G.975.1], Tables II.1 and II.2 show the achievable results for lower power implementations of the (128,119) Hamming soft decision code (with a single iteration decoder) combined with the 512 × 510 staircase code (with the hard-decision decoder described in [ITU-T G.709.2]), for QPSK and 16QAM modulations. Non-shaded rows contain measured data, while shaded rows contain extrapolated data based on simulations.

Different decoder implementations may achieve different results. For example, decoder techniques are known that achieve higher net coding gain at the expense of more power dissipation.

The net coding gain values provided below are intended as guidance. The normative application requirements are expected to be captured in a future edition of [ITU-T G.698.2].

Table II.1 – Error correcting capability of the (128,119) Hamming code plus 512 × 510 staircase code, QPSK modulation

Input BER	Output BER	Net coding gain (dB)	Coding gain (dB)	Q-limit (dB)
1.250×10^{-2}	10^{-9}	7.95	8.55	7.010
1.245×10^{-2}	10^{-10}	8.45	9.05	7.017
1.239×10^{-2}	10^{-11}	8.91	9.51	7.023
1.234×10^{-2}	10^{-12}	9.32	9.91	7.030
1.228×10^{-2}	10^{-13}	9.69	10.29	7.037
1.222×10^{-2}	10^{-14}	10.03	10.63	7.043
1.217×10^{-2}	10^{-15}	10.35	10.95	7.050

Table II.2 – Error correcting capability of the (128,119) Hamming code plus 512 × 510 staircase code, 16QAM modulation

Input BER	Output BER	Net coding gain (dB)	Coding gain (dB)
1.272×10^{-2}	10^{-9}	8.36	8.96
1.266×10^{-2}	10^{-10}	8.87	9.47
1.260×10^{-2}	10^{-11}	9.33	9.93
1.253×10^{-2}	10^{-12}	9.74	10.34
1.247×10^{-2}	10^{-13}	10.12	10.71
1.241×10^{-2}	10^{-14}	10.46	11.06
1.234×10^{-2}	10^{-15}	10.78	11.38

The Flaring threshold is $< 1 \times 10^{-22}$.

The latency of the convolutional interleaver is 7140 bits. The latency of the Hamming encoder/decoder pair is 128 bits.

Appendix III

Error correction capability of a soft decision decoder for OFEC

(This appendix does not form an integral part of this Recommendation.)

Based on the terms and definitions given in clause 7 of [ITU-T G.975.1], Tables III.1 and III.2 show the measured performance of an implementations of an OFEC soft decision decoder (with three SD iterations, 2 HD iterations and 4 bit metrics), for QPSK and 16QAM modulations.

Different decoder implementations may achieve different results. For example, some decoder techniques can achieve higher net coding gain at the expense of more power dissipation.

The net coding gain values provided below are intended as guidance. The normative application requirements are expected to be captured in a future edition of [ITU-T G.698.2].

Table III.1 – Error correcting capability of the OFEC code, QPSK modulation

Input BER	Output BER	Net coding gain (dB)	Coding gain (dB)	Q-limit (dB)
2.039×10^{-2}	10^{-9}	8.72	9.34	6.217
2.022×10^{-2}	10^{-10}	9.22	9.84	6.232
2.005×10^{-2}	10^{-11}	9.66	10.28	6.247
1.987×10^{-2}	10^{-12}	10.06	10.68	6.262
1.970×10^{-2}	10^{-13}	10.43	11.05	6.277
1.953×10^{-2}	10^{-14}	10.76	11.38	6.292
1.937×10^{-2}	10^{-15}	11.07	11.69	6.307

Table III.2 – Error correcting capability of the OFEC code, 16QAM modulation

Input BER	Output BER	Net coding gain (dB)	Coding gain (dB)
2.039×10^{-2}	10^{-9}	9.38	10.00
2.022×10^{-2}	10^{-10}	9.86	10.48
2.005×10^{-2}	10^{-11}	10.28	10.90
1.987×10^{-2}	10^{-12}	10.65	11.27
1.970×10^{-2}	10^{-13}	10.99	11.61
1.953×10^{-2}	10^{-14}	11.29	11.91
1.937×10^{-2}	10^{-15}	11.57	12.19

The Flaring threshold is $< 1 \times 10^{-22}$.

The latency of the OFEC encoder/decoder pair, including interleaving and deinterleaving, is 800,000 bits.

NOTE – 800,000 bits is ~1.6 µs (FlexO-4-DO), ~3.2 µs (FlexO-2-DO), ~6.4 µs (FlexO-1-DO).

The interleaver buffer size is 172,032 bits.

The maximum correctable burst length, when used with a hard decoder, is a traditional measure of interleaver quality. In this case, it can be shown to be 2,681 bits, which is slightly less than twice the height (in bits) of the interleaver buffer.

The OFEC code is a block-convolutional code, and its performance is characterized by its "error events." Without the " ^ (p % 16)" permutation, there are about 625,000 possible error events of weight 36 that can start at every decoding of a constituent code word. For comparison, a Product Code based on the same constituent code word has more than 3.3×10^{-13} code words of weight 36. The presence of the " ^ (p % 16)" permutation can be observed to eliminate error events of weight 36 and to raise the minimum Hamming distance of the OFEC code to at least 48.

Appendix IV

FlexO-x-DO related equation illustrations and implementation considerations

(This appendix does not form an integral part of this Recommendation.)

IV.1 Introduction

This appendix presents illustrations of equations 16-1, 16-2, 16-3 and 16-4, and bit ordering in matrices eUi, Wi, Vi and Ii.

IV.2 Illustration of equation 16-1

The j ➔ j* permutation of each 16-bit block in matrix eUi when mapped into matrix Vi is controlled by (column ^ row) and illustrated in Table IV.1.

The bit order of bits 0 to 15 in a 16-bit row block will be changed as illustrated in Table 16-6, and this change is row number dependent. For the case of row 0, the column order is unchanged. For the case of row 1, the column order of two adjacent bits is changed; e.g., 0-1 becomes 1-0. For the case of rows 2 and 3, the column order of four adjacent bits is changed; e.g., 0-1-2-3 becomes 2-3-0-1 and 3-2-1-0, etc.

Table IV.1 is a worksheet that is embedded in the table, and which can also be found in the electronic attachment to this Recommendation.

Table IV.1 – Effect of (column ^ row) permutation on the order of the bits in a 16-bit block

	column															
row	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	3	2	5	4	7	6	9	8	11	10	13	12	15	14
2	2	3	0	1	6	7	4	5	10	11	8	9	14	15	12	13
3	3	2	1	0	7	6	5	4	11	10	9	8	15	14	13	12
4	4	5	6	7	0	1	2	3	12	13	14	15	8	9	10	11
5	5	4	7	6	1	0	3	2	13	12	15	14	9	8	11	10
6	6	7	4	5	2	3	0	1	14	15	12	13	10	11	8	9
7	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
8	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
9	9	8	11	10	13	12	15	14	1	0	3	2	5	4	7	6
10	10	11	8	9	14	15	12	13	2	3	0	1	6	7	4	5
11	11	10	9	8	15	14	13	12	3	2	1	0	7	6	5	4
12	12	13	14	15	8	9	10	11	4	5	6	7	0	1	2	3
13	13	12	15	14	9	8	11	10	5	4	7	6	1	0	3	2
14	14	15	12	13	10	11	8	9	6	7	4	5	2	3	0	1
15	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The permutation of each 16-bit column in matrix Vi when mapped into Wi is controlled by (row ^ column) and is also illustrated in Table IV.1.

The bit order of bits 0 to 15 in a 16-bit column will be changed as illustrated in Table 16-6, and this change is column number dependent. For the case of column 0, the bit order is unchanged. For the case of column 1, the bit order of two adjacent bits is changed; e.g., 0-1 becomes 1-0. For the case of columns 2 and 3, the bit order of four adjacent bits is changed; e.g., 0-1-2-3 becomes 2-3-0-1 and 3-2-1-0, etc.

Table IV.2 provides a numerical illustration of equation 16-1a for {R,r} = {1,3} and of equation 16-1b for {P,p} = {0,19}. The bit permutation of the bits in matrix eUi, row {0,19} when mapping into matrix Vi, row {1,3} is illustrated by the fact that e.g., bits k = 0, 1, 2, 3 in eUi are mapped into bits {C,c} = {0,3}, {0,2}, {0,1} and {0,0} in Vi; i.e., in reverse order.

NOTE – Table IV.2 is supported by an Excel worksheet. This worksheet can be opened, after which the values of R and r as well as P and p can be modified. The worksheet is embedded in the table, and can also be found in the electronic attachment to this Recommendation.

Table IV.2 – Illustration of equations 16-1a (left) and 16-1b (right)

R	r
1	3

Vi {R,r,C,c}		eUi {P,p,k}		
C	c	P	p	k
0	0	0	19	3
0	1	0	19	2
0	2	0	19	1
0	3	0	19	0
0	4	0	19	7
0	5	0	19	6
0	6	0	19	5
0	7	0	19	4
0	8	0	19	11
0	9	0	19	10
0	10	0	19	9
0	11	0	19	8
0	12	0	19	15
0	13	0	19	14
0	14	0	19	13
0	15	0	19	12
1	0	0	19	19
1	1	0	19	18
1	2	0	19	17
1	3	0	19	16
1	4	0	19	23
1	5	0	19	22
1	6	0	19	21
1	7	0	19	20
1	8	0	19	27
1	9	0	19	26
1	10	0	19	25
1	11	0	19	24
1	12	0	19	31
1	13	0	19	30
1	14	0	19	29
1	15	0	19	28
2	0	0	19	35
2	1	0	19	34
2	2	0	19	33
2	3	0	19	32
2	4	0	19	39
2	5	0	19	38
2	6	0	19	37
2	7	0	19	36
2	8	0	19	43
2	9	0	19	42
2	10	0	19	41
2	11	0	19	40
2	12	0	19	47
2	13	0	19	46
2	14	0	19	45
2	15	0	19	44
3	0	0	19	51
3	1	0	19	50
3	2	0	19	49
3	3	0	19	48
3	4	0	19	55
3	5	0	19	54
3	6	0	19	53
3	7	0	19	52
3	8	0	19	59
3	9	0	19	58
3	10	0	19	57
3	11	0	19	56
3	12	0	19	63
3	13	0	19	62
3	14	0	19	61
3	15	0	19	60

Ii {R,r,C,c}		eUi {P,p,k}		
C	c	P	p	k
4	0	0	19	67
4	1	0	19	66
4	2	0	19	65
4	3	0	19	64
4	4	0	19	71
4	5	0	19	70
4	6	0	19	69
4	7	0	19	68
4	8	0	19	75
4	9	0	19	74
4	10	0	19	73
4	11	0	19	72
4	12	0	19	79
4	13	0	19	78
4	14	0	19	77
4	15	0	19	76
5	0	0	19	83
5	1	0	19	82
5	2	0	19	81
5	3	0	19	80
5	4	0	19	87
5	5	0	19	86
5	6	0	19	85
5	7	0	19	84
5	8	0	19	91
5	9	0	19	90
5	10	0	19	89
5	11	0	19	88
5	12	0	19	95
5	13	0	19	94
5	14	0	19	93
5	15	0	19	92
6	0	0	19	99
6	1	0	19	98
6	2	0	19	97
6	3	0	19	96
6	4	0	19	103
6	5	0	19	102
6	6	0	19	101
6	7	0	19	100
6	8	0	19	107
6	9	0	19	106
6	10	0	19	105
6	11	0	19	104
6	12	0	19	111
6	13	0	19	110
6	14	0	19	109
6	15	0	19	108
7	0	0	19	115
7	1	0	19	114
7	2	0	19	113
7	3	0	19	112
7	4	0	19	119
7	5	0	19	118
7	6	0	19	117
7	7	0	19	116
7	8	0	19	123
7	9	0	19	122
7	10	0	19	121
7	11	0	19	120
7	12	0	19	127
7	13	0	19	126
7	14	0	19	125
7	15	0	19	124

P	p
0	19

eUi {P,p,k}	Vi			
k	R	C	r	c
0	1	0	3	3
1	1	0	3	2
2	1	0	3	1
3	1	0	3	0
4	1	0	3	7
5	1	0	3	6
6	1	0	3	5
7	1	0	3	4
8	1	0	3	11
9	1	0	3	10
10	1	0	3	9
11	1	0	3	8
12	1	0	3	15
13	1	0	3	14
14	1	0	3	13
15	1	0	3	12
16	1	1	3	3
17	1	1	3	2
18	1	1	3	1
19	1	1	3	0
20	1	1	3	7
21	1	1	3	6
22	1	1	3	5
23	1	1	3	4
24	1	1	3	11
25	1	1	3	10
26	1	1	3	9
27	1	1	3	8
28	1	1	3	15
29	1	1	3	14
30	1	1	3	13
31	1	1	3	12
32	1	2	3	3
33	1	2	3	2
34	1	2	3	1
35	1	2	3	0
36	1	2	3	7
37	1	2	3	6
38	1	2	3	5
39	1	2	3	4
40	1	2	3	11
41	1	2	3	10
42	1	2	3	9
43	1	2	3	8
44	1	2	3	15
45	1	2	3	14
46	1	2	3	13
47	1	2	3	12
48	1	3	3	3
49	1	3	3	2
50	1	3	3	1
51	1	3	3	0
52	1	3	3	7
53	1	3	3	6
54	1	3	3	5
55	1	3	3	4
56	1	3	3	11
57	1	3	3	10
58	1	3	3	9
59	1	3	3	8
60	1	3	3	15
61	1	3	3	14
62	1	3	3	13
63	1	3	3	12

eUi {P,p,k}	Vi			
k	R	C	r	c
64	1	4	3	3
65	1	4	3	2
66	1	4	3	1
67	1	4	3	0
68	1	4	3	7
69	1	4	3	6
70	1	4	3	5
71	1	4	3	4
72	1	4	3	11
73	1	4	3	10
74	1	4	3	9
75	1	4	3	8
76	1	4	3	15
77	1	4	3	14
78	1	4	3	13
79	1	4	3	12
80	1	5	3	3
81	1	5	3	2
82	1	5	3	1
83	1	5	3	0
84	1	5	3	7
85	1	5	3	6
86	1	5	3	5
87	1	5	3	4
88	1	5	3	11
89	1	5	3	10
90	1	5	3	9
91	1	5	3	8
92	1	5	3	15
93	1	5	3	14
94	1	5	3	13
95	1	5	3	12
96	1	6	3	3
97	1	6	3	2
98	1	6	3	1
99	1	6	3	0
100	1	6	3	7
101	1	6	3	6
102	1	6	3	5
103	1	6	3	4
104	1	6	3	11
105	1	6	3	10
106	1	6	3	9
107	1	6	3	8
108	1	6	3	15
109	1	6	3	14
110	1	6	3	13
111	1	6	3	12
112	1	7	3	3
113	1	7	3	2
114	1	7	3	1
115	1	7	3	0
116	1	7	3	7
117	1	7	3	6
118	1	7	3	5
119	1	7	3	4
120	1	7	3	11
121	1	7	3	10
122	1	7	3	9
123	1	7	3	8
124	1	7	3	15
125	1	7	3	14
126	1	7	3	13
127	1	7	3	12

IV.3 Illustration of equations 16-2 and 16-3

Table IV.3 provides a numerical illustration of Wi(t,P,p,k) for P = 9 and p = 20 and Z = 8. The bit permutation of the bits in matrix Vi, column 4 when mapping into Wi is illustrated by the fact that bit k = 0, 16, 32, … is derived from the bit in Vi, row 4 and that bit k = 1, 17, 33, … is derived from the bit in Vi, row 5 and that bit k = 4, 20, 36, … is derived from the bit in Vi, row 0, etc.

NOTE – Table IV.3 is supported by an Excel worksheet. This worksheet can be opened, after which the values of P, p and Z can be modified. The worksheet is embedded in the table, and can also be found in the electronic attachment to this Recommendation.

Table IV.3 – Wi(t,P,p,k) as a function of Vi{t,R,C,r,c} and Ui{P,p,k}

Z	P	p
8	9	20

Vi {P,p, k }	Vi {t	R	C	r	c}	Vi {P,p, k }	Vi {t	R	C	r	c}	Vi {P,p, k }	eUi {P	p	k}	Vi {P,p, k }	eUi {P	p	k}
0	t-1	334	0	4	4	64	t	6	4	4	4	128	9	20	0	192	9	20	64
1	t-1	334	0	5	4	65	t	6	4	5	4	129	9	20	1	193	9	20	65
2	t-1	334	0	6	4	66	t	6	4	6	4	130	9	20	2	194	9	20	66
3	t-1	334	0	7	4	67	t	6	4	7	4	131	9	20	3	195	9	20	67
4	t-1	334	0	0	4	68	t	6	4	0	4	132	9	20	4	196	9	20	68
5	t-1	334	0	1	4	69	t	6	4	1	4	133	9	20	5	197	9	20	69
6	t-1	334	0	2	4	70	t	6	4	2	4	134	9	20	6	198	9	20	70
7	t-1	334	0	3	4	71	t	6	4	3	4	135	9	20	7	199	9	20	71
8	t-1	334	0	12	4	72	t	6	4	12	4	136	9	20	8	200	9	20	72
9	t-1	334	0	13	4	73	t	6	4	13	4	137	9	20	9	201	9	20	73
10	t-1	334	0	14	4	74	t	6	4	14	4	138	9	20	10	202	9	20	74
11	t-1	334	0	15	4	75	t	6	4	15	4	139	9	20	11	203	9	20	75
12	t-1	334	0	8	4	76	t	6	4	8	4	140	9	20	12	204	9	20	76
13	t-1	334	0	9	4	77	t	6	4	9	4	141	9	20	13	205	9	20	77
14	t-1	334	0	10	4	78	t	6	4	10	4	142	9	20	14	206	9	20	78
15	t-1	334	0	11	4	79	t	6	4	11	4	143	9	20	15	207	9	20	79
16	t	0	1	4	4	80	t	8	5	4	4	144	9	20	16	208	9	20	80
17	t	0	1	5	4	81	t	8	5	5	4	145	9	20	17	209	9	20	81
18	t	0	1	6	4	82	t	8	5	6	4	146	9	20	18	210	9	20	82
19	t	0	1	7	4	83	t	8	5	7	4	147	9	20	19	211	9	20	83
20	t	0	1	0	4	84	t	8	5	0	4	148	9	20	20	212	9	20	84
21	t	0	1	1	4	85	t	8	5	1	4	149	9	20	21	213	9	20	85
22	t	0	1	2	4	86	t	8	5	2	4	150	9	20	22	214	9	20	86
23	t	0	1	3	4	87	t	8	5	3	4	151	9	20	23	215	9	20	87
24	t	0	1	12	4	88	t	8	5	12	4	152	9	20	24	216	9	20	88
25	t	0	1	13	4	89	t	8	5	13	4	153	9	20	25	217	9	20	89
26	t	0	1	14	4	90	t	8	5	14	4	154	9	20	26	218	9	20	90
27	t	0	1	15	4	91	t	8	5	15	4	155	9	20	27	219	9	20	91
28	t	0	1	8	4	92	t	8	5	8	4	156	9	20	28	220	9	20	92
29	t	0	1	9	4	93	t	8	5	9	4	157	9	20	29	221	9	20	93
30	t	0	1	10	4	94	t	8	5	10	4	158	9	20	30	222	9	20	94
31	t	0	1	11	4	95	t	8	5	11	4	159	9	20	31	223	9	20	95
32	t	2	2	4	4	96	t	10	6	4	4	160	9	20	32	224	9	20	96
33	t	2	2	5	4	97	t	10	6	5	4	161	9	20	33	225	9	20	97
34	t	2	2	6	4	98	t	10	6	6	4	162	9	20	34	226	9	20	98
35	t	2	2	7	4	99	t	10	6	7	4	163	9	20	35	227	9	20	99
36	t	2	2	0	4	100	t	10	6	0	4	164	9	20	36	228	9	20	100
37	t	2	2	1	4	101	t	10	6	1	4	165	9	20	37	229	9	20	101
38	t	2	2	2	4	102	t	10	6	2	4	166	9	20	38	230	9	20	102
39	t	2	2	3	4	103	t	10	6	3	4	167	9	20	39	231	9	20	103
40	t	2	2	12	4	104	t	10	6	12	4	168	9	20	40	232	9	20	104
41	t	2	2	13	4	105	t	10	6	13	4	169	9	20	41	233	9	20	105
42	t	2	2	14	4	106	t	10	6	14	4	170	9	20	42	234	9	20	106
43	t	2	2	15	4	107	t	10	6	15	4	171	9	20	43	235	9	20	107
44	t	2	2	8	4	108	t	10	6	8	4	172	9	20	44	236	9	20	108
45	t	2	2	9	4	109	t	10	6	9	4	173	9	20	45	237	9	20	109
46	t	2	2	10	4	110	t	10	6	10	4	174	9	20	46	238	9	20	110
47	t	2	2	11	4	111	t	10	6	11	4	175	9	20	47	239			
48	t	4	3	4	4	112	t	12	7	4	4	176	9	20	48	240			
49	t	4	3	5	4	113	t	12	7	5	4	177	9	20	49	241			
50	t	4	3	6	4	114	t	12	7	6	4	178	9	20	50	242			
51	t	4	3	7	4	115	t	12	7	7	4	179	9	20	51	243			
52	t	4	3	0	4	116	t	12	7	0	4	180	9	20	52	244			
53	t	4	3	1	4	117	t	12	7	1	4	181	9	20	53	245			
54	t	4	3	2	4	118	t	12	7	2	4	182	9	20	54	246			
55	t	4	3	3	4	119	t	12	7	3	4	183	9	20	55	247			
56	t	4	3	12	4	120	t	12	7	12	4	184	9	20	56	248			
57	t	4	3	13	4	121	t	12	7	13	4	185	9	20	57	249			
58	t	4	3	14	4	122	t	12	7	14	4	186	9	20	58	250			
59	t	4	3	15	4	123	t	12	7	15	4	187	9	20	59	251			
60	t	4	3	8	4	124	t	12	7	8	4	188	9	20	60	252			
61	t	4	3	9	4	125	t	12	7	9	4	189	9	20	61	253			
62	t	4	3	10	4	126	t	12	7	10	4	190	9	20	62	254			
63	t	4	3	11	4	127	t	12	7	11	4	191	9	20	63	255			

IV.4 Illustration of equation 16-4

The Vi{R,C} ➔ Ii{R,C}* permutation in each 16×16-bit block {R,C} is controlled by:

$$Ii\{r,c\} = Vi\{\ r = ((c - 2r - \lfloor r/8 \rfloor)\ \%\ 16)\ ,\ c = ((c - r - \lfloor r/8 \rfloor)\ \%\ 16)\ \} \qquad \text{(IV-1)}$$

The (row,column) order of bits (0,0) to (15,15) in such 16 × 16-bit block will be changed as illustrated in Table IV.4.

For example, the bit in Ii{r = 0 , c = 1} contains the bit from Vi{ r = 1 , c = 1} and the bit in Ii{r = 12, c = 8} contains the bit from Vi{ r = 15 , c = 11}.

NOTE – The left entries of the pairs in Table IV.3 form a Latin Square. The right entries almost form a Latin square, but they are duplicated in the first and last rows. Table IV.1 is a worksheet that is embedded in the table, and which can also be found in the electronic attachment to this Recommendation.

Table IV.4 – Effect of permutation on a 16×16-bit block {R,C} in matrix Vi when mapped into matrix Ii

r\c	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0,0	1,1	2,2	3,3	4,4	5,5	6,6	7,7	8,8	9,9	10,10	11,11	12,12	13,13	14,14	15,15
1	14,15	15,0	0,1	1,2	2,3	3,4	4,5	5,6	6,7	7,8	8,9	9,10	10,11	11,12	12,13	13,14
2	12,14	13,15	14,0	15,1	0,2	1,3	2,4	3,5	4,6	5,7	6,8	7,9	8,10	9,11	10,12	11,13
3	10,13	11,14	12,15	13,0	14,1	15,2	0,3	1,4	2,5	3,6	4,7	5,8	6,9	7,10	8,11	9,12
4	8,12	9,13	10,14	11,15	12,0	13,1	14,2	15,3	0,4	1,5	2,6	3,7	4,8	5,9	6,10	7,11
5	6,11	7,12	8,13	9,14	10,15	11,0	12,1	13,2	14,3	15,4	0,5	1,6	2,7	3,8	4,9	5,10
6	4,10	5,11	6,12	7,13	8,14	9,15	10,0	11,1	12,2	13,3	14,4	15,5	0,6	1,7	2,8	3,9
7	2,9	3,10	4,11	5,12	6,13	7,14	8,15	9,0	10,1	11,2	12,3	13,4	14,5	15,6	0,7	1,8
8	15,7	0,8	1,9	2,10	3,11	4,12	5,13	6,14	7,15	8,0	9,1	10,2	11,3	12,4	13,5	14,6
9	13,6	14,7	15,8	0,9	1,10	2,11	3,12	4,13	5,14	6,15	7,0	8,1	9,2	10,3	11,4	12,5
10	11,5	12,6	13,7	14,8	15,9	0,10	1,11	2,12	3,13	4,14	5,15	6,0	7,1	8,2	9,3	10,4
11	9,4	10,5	11,6	12,7	13,8	14,9	15,10	0,11	1,12	2,13	3,14	4,15	5,0	6,1	7,2	8,3
12	7,3	8,4	9,5	10,6	11,7	12,8	13,9	14,10	15,11	0,12	1,13	2,14	3,15	4,0	5,1	6,2
13	5,2	6,3	7,4	8,5	9,6	10,7	11,8	12,9	13,10	14,11	15,12	0,13	1,14	2,15	3,0	4,1
14	3,1	4,2	5,3	6,4	7,5	8,6	9,7	10,8	11,9	12,10	13,11	14,12	15,13	0,14	1,15	2,0
15	1,0	2,1	3,2	4,3	5,4	6,5	7,6	8,7	9,8	10,9	11,10	12,11	13,12	14,13	15,14	0,15

IV.5 Illustration of bit ordering in eUi, Wi, Vi and Ii

Table IV.5 illustrates, from top to bottom, the location of OFC input bits 0 to 7103 and associated parity bits 7104 to 8190 for P = 10, p = 16 and i = 0 in Row P of matrix eUi, Wi(P,p), Rows R = 2P, 2P+1 of matrix Vi and Rows R = 2P, 2P+1 of matrix Ii.

NOTE – Table IV.5 is supported by an Excel worksheet. This worksheet can be opened, after which the values of P, p and i can be modified. The worksheet is embedded in the figure, and can also be found in the electronic attachment to this Recommendation.

Table IV.5 – Illustration of location of OFC input bits and associated parity bits in eUi, Wi, Vi and Ii

P	p	i																																
			matrix eUi (i=0,1), row P					OFC input bits are numbered 0 to7103 and parity bits are numbered 7104 to 8190												fields containing parity bits are colored green														
10	16	0	matrix eUi,Row P																															
eUi	P	p\k	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*	*30*	*31*
	10	*0*	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	1024	1026	1028	1030	1032	1034	1036	1038	1040	1042	1044	1046	1048	1050	1052	1054
	10	*1*	32	34	36	38	40	42	44	46	48	50	52	54	56	58	60	62	1056	1058	1060	1062	1064	1066	1068	1070	1072	1074	1076	1078	1080	1082	1084	1086
	10	*2*	64	66	68	70	72	74	76	78	80	82	84	86	88	90	92	94	1088	1090	1092	1094	1096	1098	1100	1102	1104	1106	1108	1110	1112	1114	1116	1118
	10	*3*	96	98	100	102	104	106	108	110	112	114	116	118	120	122	124	126	1120	1122	1124	1126	1128	1130	1132	1134	1136	1138	1140	1142	1144	1146	1148	1150
	10	*4*	128	130	132	134	136	138	140	142	144	146	148	150	152	154	156	158	1152	1154	1156	1158	1160	1162	1164	1166	1168	1170	1172	1174	1176	1178	1180	1182
	10	*5*	160	162	164	166	168	170	172	174	176	178	180	182	184	186	188	190	1184	1186	1188	1190	1192	1194	1196	1198	1200	1202	1204	1206	1208	1210	1212	1214
	10	*6*	192	194	196	198	200	202	204	206	208	210	212	214	216	218	220	222	1216	1218	1220	1222	1224	1226	1228	1230	1232	1234	1236	1238	1240	1242	1244	1246
	10	*7*	224	226	228	230	232	234	236	238	240	242	244	246	248	250	252	254	1248	1250	1252	1254	1256	1258	1260	1262	1264	1266	1268	1270	1272	1274	1276	1278
	10	*8*	256	258	260	262	264	266	268	270	272	274	276	278	280	282	284	286	1280	1282	1284	1286	1288	1290	1292	1294	1296	1298	1300	1302	1304	1306	1308	1310
	10	*9*	288	290	292	294	296	298	300	302	304	306	308	310	312	314	316	318	1312	1314	1316	1318	1320	1322	1324	1326	1328	1330	1332	1334	1336	1338	1340	1342
	10	*10*	320	322	324	326	328	330	332	334	336	338	340	342	344	346	348	350	1344	1346	1348	1350	1352	1354	1356	1358	1360	1362	1364	1366	1368	1370	1372	1374
	10	*11*	352	354	356	358	360	362	364	366	368	370	372	374	376	378	380	382	1376	1378	1380	1382	1384	1386	1388	1390	1392	1394	1396	1398	1400	1402	1404	1406
	10	*12*	384	386	388	390	392	394	396	398	400	402	404	406	408	410	412	414	1408	1410	1412	1414	1416	1418	1420	1422	1424	1426	1428	1430	1432	1434	1436	1438
	10	*13*	416	418	420	422	424	426	428	430	432	434	436	438	440	442	444	446	1440	1442	1444	1446	1448	1450	1452	1454	1456	1458	1460	1462	1464	1466	1468	1470
	10	*14*	448	450	452	454	456	458	460	462	464	466	468	470	472	474	476	478	1472	1474	1476	1478	1480	1482	1484	1486	1488	1490	1492	1494	1496	1498	1500	1502
	10	*15*	480	482	484	486	488	490	492	494	496	498	500	502	504	506	508	510	1504	1506	1508	1510	1512	1514	1516	1518	1520	1522	1524	1526	1528	1530	1532	1534
	10	*16*	512	514	516	518	520	522	524	526	528	530	532	534	536	538	540	542	1536	1538	1540	1542	1544	1546	1548	1550	1552	1554	1556	1558	1560	1562	1564	1566
	10	*17*	544	546	548	550	552	554	556	558	560	562	564	566	568	570	572	574	1568	1570	1572	1574	1576	1578	1580	1582	1584	1586	1588	1590	1592	1594	1596	1598
	10	*18*	576	578	580	582	584	586	588	590	592	594	596	598	600	602	604	606	1600	1602	1604	1606	1608	1610	1612	1614	1616	1618	1620	1622	1624	1626	1628	1630
	10	*19*	608	610	612	614	616	618	620	622	624	626	628	630	632	634	636	638	1632	1634	1636	1638	1640	1642	1644	1646	1648	1650	1652	1654	1656	1658	1660	1662
	10	*20*	640	642	644	646	648	650	652	654	656	658	660	662	664	666	668	670	1664	1666	1668	1670	1672	1674	1676	1678	1680	1682	1684	1686	1688	1690	1692	1694
	10	*21*	672	674	676	678	680	682	684	686	688	690	692	694	696	698	700	702	1696	1698	1700	1702	1704	1706	1708	1710	1712	1714	1716	1718	1720	1722	1724	1726
	10	*22*	704	706	708	710	712	714	716	718	720	722	724	726	728	730	732	734	1728	1730	1732	1734	1736	1738	1740	1742	1744	1746	1748	1750	1752	1754	1756	1758
	10	*23*	736	738	740	742	744	746	748	750	752	754	756	758	760	762	764	766	1760	1762	1764	1766	1768	1770	1772	1774	1776	1778	1780	1782	1784	1786	1788	1790
	10	*24*	768	770	772	774	776	778	780	782	784	786	788	790	792	794	796	798	1792	1794	1796	1798	1800	1802	1804	1806	1808	1810	1812	1814	1816	1818	1820	1822
	10	*25*	800	802	804	806	808	810	812	814	816	818	820	822	824	826	828	830	1824	1826	1828	1830	1832	1834	1836	1838	1840	1842	1844	1846	1848	1850	1852	1854
	10	*26*	832	834	836	838	840	842	844	846	848	850	852	854	856	858	860	862	1856	1858	1860	1862	1864	1866	1868	1870	1872	1874	1876	1878	1880	1882	1884	1886
	10	*27*	864	866	868	870	872	874	876	878	880	882	884	886	888	890	892	894	1888	1890	1892	1894	1896	1898	1900	1902	1904	1906	1908	1910	1912	1914	1916	1918
	10	*28*	896	898	900	902	904	906	908	910	912	914	916	918	920	922	924	926	1920	1922	1924	1926	1928	1930	1932	1934	1936	1938	1940	1942	1944	1946	1948	1950
	10	*29*	928	930	932	934	936	938	940	942	944	946	948	950	952	954	956	958	1952	1954	1956	1958	1960	1962	1964	1966	1968	1970	1972	1974	1976	1978	1980	1982
	10	*30*	960	962	964	966	968	970	972	974	976	978	980	982	984	986	988	990	1984	1986	1988	1990	1992	1994	1996	1998	2000	2002	2004	2006	2008	2010	2012	2014
	10	*31*	992	994	996	998	1000	1002	1004	1006	1008	1010	1012	1014	1016	1018	1020	1022	2016	2018	2020	2022	2024	2026	2028	2030	2032	2034	2036	2038	2040	2042	2044	2046

		Wi(P,p)				Wi(P,p,k, k = 0..127) = Vi { ((2P + [p/16]) ^ 1) – 20 + 2 × [k/16] , [k/16] , (k % 16) ^ (p % 16) , (p % 16)																		Wi(P,p,k, k = 128..238) = Ui { P, p, k – 128 }									
Wi(P,p)	k	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*	*25*	*26*	*27*	*28*	*29*	*30*	*31*
Vi, row R		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Vi row R, bit		0	34	68	102	136	170	204	238	272	306	340	374	408	442	476	510	1024	1058	1092	1126	1160	1194	1228	1262	1296	1330	1364	1398	1432	1466	1500	1534
eUi row P, bit		512	514	516	518	520	522	524	526	528	530	532	534	536	538	540	542	1536	1538	1540	1542	1544	1546	1548	1550	1552	1554	1556	1558	1560	1562	1564	1566
	k	*128*	*129*	*130*	*131*	*132*	*133*	*134*	*135*	*136*	*137*	*138*	*139*	*140*	*141*	*142*	*143*	*144*	*145*	*146*	*147*	*148*	*149*	*150*	*151*	*152*	*153*	*154*	*155*	*156*	*157*	*158*	*159*

		matrix Vi, Rows R = 2P and R = 2P+1						Vi {R,C,r,c} = eUi {P = [R/2] , p = (R % 2) × 16 + r , k = 16 × C + (r ^c)}																								
Vi	C	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
R	r\c	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
20	*0*	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	1024	1026	1028	1030	1032	1034	1036	1038	1040	1042	1044	1046	1048	1050	1052	1054
20	*1*	34	32	38	36	42	40	46	44	50	48	54	52	58	56	62	60	1058	1056	1062	1060	1066	1064	1070	1068	1074	1072	1078	1076	1082	1080	1086	1084
20	*2*	68	70	64	66	76	78	72	74	84	86	80	82	92	94	88	90	1092	1094	1088	1090	1100	1102	1096	1098	1108	1110	1104	1106	1116	1118	1112	1114
20	*3*	102	100	98	96	110	108	106	104	118	116	114	112	126	124	122	120	1126	1124	1122	1120	1134	1132	1130	1128	1142	1140	1138	1136	1150	1148	1146	1144
20	*4*	136	138	140	142	128	130	132	134	152	154	156	158	144	146	148	150	1160	1162	1164	1166	1152	1154	1156	1158	1176	1178	1180	1182	1168	1170	1172	1174
20	*5*	170	168	174	172	162	160	166	164	186	184	190	188	178	176	182	180	1194	1192	1198	1196	1186	1184	1190	1188	1210	1208	1214	1212	1202	1200	1206	1204
20	*6*	204	206	200	202	196	198	192	194	220	222	216	218	212	214	208	210	1228	1230	1224	1226	1220	1222	1216	1218	1244	1246	1240	1242	1236	1238	1232	1234
20	*7*	238	236	234	232	230	228	226	224	254	252	250	248	246	244	242	240	1262	1260	1258	1256	1254	1252	1250	1248	1278	1276	1274	1272	1270	1268	1266	1264
20	*8*	272	274	276	278	280	282	284	286	256	258	260	262	264	266	268	270	1296	1298	1300	1302	1304	1306	1308	1310	1280	1282	1284	1286	1288	1290	1292	1294
20	*9*	306	304	310	308	314	312	318	316	290	288	294	292	298	296	302	300	1330	1328	1334	1332	1338	1336	1342	1340	1314	1312	1318	1316	1322	1320	1326	1324
20	*10*	340	342	336	338	348	350	344	346	324	326	320	322	332	334	328	330	1364	1366	1360	1362	1372	1374	1368	1370	1348	1350	1344	1346	1356	1358	1352	1354
20	*11*	374	372	370	368	382	380	378	376	358	356	354	352	366	364	362	360	1398	1396	1394	1392	1406	1404	1402	1400	1382	1380	1378	1376	1390	1388	1386	1384
20	*12*	408	410	412	414	400	402	404	406	392	394	396	398	384	386	388	390	1432	1434	1436	1438	1424	1426	1428	1430	1416	1418	1420	1422	1408	1410	1412	1414
20	*13*	442	440	446	444	434	432	438	436	426	424	430	428	418	416	422	420	1466	1464	1470	1468	1458	1456	1462	1460	1450	1448	1454	1452	1442	1440	1446	1444
20	*14*	476	478	472	474	468	470	464	466	460	462	456	458	452	454	448	450	1500	1502	1496	1498	1492	1494	1488	1490	1484	1486	1480	1482	1476	1478	1472	1474
20	*15*	510	508	506	504	502	500	498	496	494	492	490	488	486	484	482	480	1534	1532	1530	1528	1526	1524	1522	1520	1518	1516	1514	1512	1510	1508	1506	1504
21	*0*	512	514	516	518	520	522	524	526	528	530	532	534	536	538	540	542	1536	1538	1540	1542	1544	1546	1548	1550	1552	1554	1556	1558	1560	1562	1564	1566
21	*1*	546	544	550	548	554	552	558	556	562	560	566	564	570	568	574	572	1570	1568	1574	1572	1578	1576	1582	1580	1586	1584	1590	1588	1594	1592	1598	1596
21	*2*	580	582	576	578	588	590	584	586	596	598	592	594	604	606	600	602	1604	1606	1600	1602	1612	1614	1608	1610	1620	1622	1616	1618	1628	1630	1624	1626
21	*3*	614	612	610	608	622	620	618	616	630	628	626	624	638	636	634	632	1638	1636	1634	1632	1646	1644	1642	1640	1654	1652	1650	1648	1662	1660	1658	1656
21	*4*	648	650	652	654	640	642	644	646	664	666	668	670	656	658	660	662	1672	1674	1676	1678	1664	1666	1668	1670	1688	1690	1692	1694	1680	1682	1684	1686
21	*5*	682	680	686	684	674	672	678	676	698	696	702	700	690	688	694	692	1706	1704	1710	1708	1698	1696	1702	1700	1722	1720	1726	1724	1714	1712	1718	1716
21	*6*	716	718	712	714	708	710	704	706	732	734	728	730	724	726	720	722	1740	1742	1736	1738	1732	1734	1728	1730	1756	1758	1752	1754	1748	1750	1744	1746
21	*7*	750	748	746	744	742	740	738	736	766	764	762	760	758	756	754	752	1774	1772	1770	1768	1766	1764	1762	1760	1790	1788	1786	1784	1782	1780	1778	1776
21	*8*	784	786	788	790	792	794	796	798	768	770	772	774	776	778	780	782	1808	1810	1812	1814	1816	1818	1820	1822	1792	1794	1796	1798	1800	1802	1804	1806
21	*9*	818	816	822	820	826	824	830	828	802	800	806	804	810	808	814	812	1842	1840	1846	1844	1850	1848	1854	1852	1826	1824	1830	1828	1834	1832	1838	1836
21	*10*	852	854	848	850	860	862	856	858	836	838	832	834	844	846	840	842	1876	1878	1872	1874	1884	1886	1880	1882	1860	1862	1856	1858	1868	1870	1864	1866
21	*11*	886	884	882	880	894	892	890	888	870	868	866	864	878	876	874	872	1910	1908	1906	1904	1918	1916	1914	1912	1894	1892	1890	1888	1902	1900	1898	1896
21	*12*	920	922	924	926	912	914	916	918	904	906	908	910	896	898	900	902	1944	1946	1948	1950	1936	1938	1940	1942	1928	1930	1932	1934	1920	1922	1924	1926
21	*13*	954	952	958	956	946	944	950	948	938	936	942	940	930	928	934	932	1978	1976	1982	1980	1970	1968	1974	1972	1962	1960	1966	1964	1954	1952	1958	1956
21	*14*	988	990	984	986	980	982	976	978	972	974	968	970	964	966	960	962	2012	2014	2008	2010	2004	2006	2000	2002	1996	1998	1992	1994	1988	1990	1984	1986
21	*15*	1022	1020	1018	1016	1014	1012	1010	1008	1006	1004	1002	1000	998	996	994	992	2046	2044	2042	2040	2038	2036	2034	2032	2030	2028	2026	2024	2022	2020	2018	2016

		matrix Ii, Rows R = 2P and R = 2P+1; with marking of 8-bit mapping blocks												Ii{ R , C , r ,c } = Vi{ R , C , ((c-2r-[r/8])%16) , ((c-r-[r/8])%16) }																				
Ii	C	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*
R	r\c	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*	0	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
20	*0*	0	32	64	96	128	160	192	224	256	288	320	352	384	416	448	480	1024	1056	1088	1120	1152	1184	1216	1248	1280	1312	1344	1376	1408	1440	1472	1504
20	*1*	450	510	2	38	66	110	130	166	194	254	258	294	322	366	386	422	1474	1534	1026	1062	1090	1134	1154	1190	1218	1278	1282	1318	1346	1390	1410	1446
20	*2*	388	420	476	508	4	36	76	108	132	164	220	252	260	292	332	364	1412	1444	1500	1532	1028	1060	1100	1132	1156	1188	1244	1276	1284	1316	1356	1388
20	*3*	334	362	390	442	478	506	6	42	78	106	134	186	222	250	262	298	1358	1386	1414	1466	1502	1530	1030	1066	1102	1130	1158	1210	1246	1274	1286	1322
20	*4*	264	296	328	360	408	440	472	504	8	40	72	104	152	184	216	248	1288	1320	1352	1384	1432	1464	1496	1528	1032	1064	1096	1128	1176	1208	1240	1272
20	*5*	218	246	266	302	330	374	410	446	474	502	10	46	74	118	154	190	1242	1270	1290	1326	1354	1398	1434	1470	1498	1526	1034	1070	1098	1142	1178	1214
20	*6*	156	188	212	244	268	300	340	372	412	444	468	500	12	44	84	116	1180	1212	1236	1268	1292	1324	1364	1396	1436	1468	1492	1524	1036	1068	1108	1140
20	*7*	86	114	158	178	214	242	270	306	342	370	414	434	470	498	14	50	1110	1138	1182	1202	1238	1266	1294	1330	1366	1394	1438	1458	1494	1522	1038	1074
20	*8*	496	16	48	80	112	144	176	208	240	272	304	336	368	400	432	464	1520	1040	1072	1104	1136	1168	1200	1232	1264	1296	1328	1360	1392	1424	1456	1488
20	*9*	438	466	494	18	54	82	126	146	182	210	238	274	310	338	382	402	1462	1490	1518	1042	1078	1106	1150	1170	1206	1234	1262	1298	1334	1362	1406	1426
20	*10*	380	404	436	460	492	20	52	92	124	148	180	204	236	276	308	348	1404	1428	1460	1484	1516	1044	1076	1116	1148	1172	1204	1228	1260	1300	1332	1372
20	*11*	314	350	378	406	426	462	490	22	58	94	122	150	170	206	234	278	1338	1374	1402	1430	1450	1486	1514	1046	1082	1118	1146	1174	1194	1230	1258	1302
20	*12*	232	280	312	344	376	392	424	456	488	24	56	88	120	136	168	200	1256	1304	1336	1368	1400	1416	1448	1480	1512	1048	1080	1112	1144	1160	1192	1224
20	*13*	174	202	230	282	318	346	358	394	430	458	486	26	62	90	102	138	1198	1226	1254	1306	1342	1370	1382	1418	1454	1482	1510	1050	1086	1114	1126	1162
20	*14*	100	140	172	196	228	284	316	324	356	396	428	452	484	28	60	68	1124	1164	1196	1220	1252	1308	1340	1348	1380	1420	1452	1476	1508	1052	1084	1092
20	*15*	34	70	98	142	162	198	226	286	290	326	354	398	418	454	482	30	1058	1094	1122	1166	1186	1222	1250	1310	1314	1350	1378	1422	1442	1478	1506	1054
21	*0*	512	544	576	608	640	672	704	736	768	800	832	864	896	928	960	992	1536	1568	1600	1632	1664	1696	1728	1760	1792	1824	1856	1888	1920	1952	1984	2016
21	*1*	962	1022	514	550	578	622	642	678	706	766	770	806	834	878	898	934	1986	2046	1538	1574	1602	1646	1666	1702	1730	1790	1794	1830	1858	1902	1922	1958
21	*2*	900	932	988	1020	516	548	588	620	644	676	732	764	772	804	844	876	1924	1956	2012	2044	1540	1572	1612	1644	1668	1700	1756	1788	1796	1828	1868	1900
21	*3*	846	874	902	954	990	1018	518	554	590	618	646	698	734	762	774	810	1870	1898	1926	1978	2014	2042	1542	1578	1614	1642	1670	1722	1758	1786	1798	1834
21	*4*	776	808	840	872	920	952	984	1016	520	552	584	616	664	696	728	760	1800	1832	1864	1896	1944	1976	2008	2040	1544	1576	1608	1640	1688	1720	1752	1784
21	*5*	730	758	778	814	842	886	922	958	986	1014	522	558	586	630	666	702	1754	1782	1802	1838	1866	1910	1946	1982	2010	2038	1546	1582	1610	1654	1690	1726
21	*6*	668	700	724	756	780	812	852	884	924	956	980	1012	524	556	596	628	1692	1724	1748	1780	1804	1836	1876	1908	1948	1980	2004	2036	1548	1580	1620	1652
21	*7*	598	626	670	690	726	754	782	818	854	882	926	946	982	1010	526	562	1622	1650	1694	1714	1750	1778	1806	1842	1878	1906	1950	1970	2006	2034	1550	1586
21	*8*	1008	528	560	592	624	656	688	720	752	784	816	848	880	912	944	976	2032	1552	1584	1616	1648	1680	1712	1744	1776	1808	1840	1872	1904	1936	1968	2000
21	*9*	950	978	1006	530	566	594	638	658	694	722	750	786	822	850	894	914	1974	2002	2030	1554	1590	1618	1662	1682	1718	1746	1774	1810	1846	1874	1918	1938
21	*10*	892	916	948	972	1004	532	564	604	636	660	692	716	748	788	820	860	1916	1940	1972	1996	2028	1556	1588	1628	1660	1684	1716	1740	1772	1812	1844	1884
21	*11*	826	862	890	918	938	974	1002	534	570	606	634	662	682	718	746	790	1850	1886	1914	1942	1962	1998	2026	1558	1594	1630	1658	1686	1706	1742	1770	1814
21	*12*	744	792	824	856	888	904	936	968	1000	536	568	600	632	648	680	712	1768	1816	1848	1880	1912	1928	1960	1992	2024	1560	1592	1624	1656	1672	1704	1736
21	*13*	686	714	742	794	830	858	870	906	942	970	998	538	574	602	614	650	1710	1738	1766	1818	1854	1882	1894	1930	1966	1994	2022	1562	1598	1626	1638	1674
21	*14*	612	652	684	708	740	796	828	836	868	908	940	964	996	540	572	580	1636	1676	1708	1732	1764	1820	1852	1860	1892	1932	1964	1988	2020	1564	1596	1604
21	*15*	546	582	610	654	674	710	738	798	802	838	866	910	930	966	994	542	1570	1606	1634	1678	1698	1734	1762	1822	1826	1862	1890	1934	1954	1990	2018	1566

Appendix V

Generic principles of forward error correction using blockwise-recursively-encoded open FEC

(This appendix does not form an integral part of this Recommendation.)

V.1 Open FEC codes: Specifications and basic properties

Open FEC (O FEC) codes are a class of error-correcting codes that combine ideas from recursive convolutional coding and block coding, resulting in a "continuous" product-like code that is characterized by the relationship between successive matrices of bits. They are closely related to Braided Codes. Refer to [b-Tanner], [b-Zigangirov] and [b-Feltström].

Consider the (semi-infinite) rectangle of $m \times m$ matrices $B_{i,j}$, for $i,j \in \mathbb{Z}^+$, where the elements of $B_{i,j}$ are binary (i.e., in $GF(2)$) and where each row has $T = n/m$ matrices, as illustrated in Figure V.1.

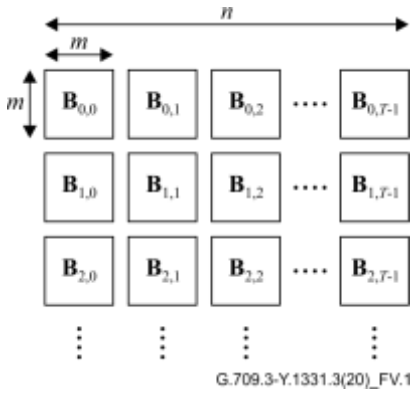


Figure V.1 – A stream of rectangle $m \times m$ arrays of bits

First, a conventional FEC block code (e.g., SPC, Hamming, BCH, RS) in systematic form is selected to serve as the constituent code; this code, referred to as W, is selected to have block length $2n$ bits, r of which are parity bits and $2n - r$ of which are information bits. As illustrated in Figure V.2, the leftmost $2n - r$ bits constitute information positions of W, and the rightmost r bit constitute the parity positions of W.

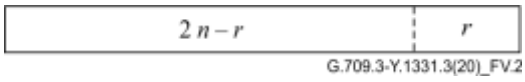


Figure V.2 – Subdivision of the length $2n$ systematic constituent code word into its leftmost $2n - r$ information bits and its rightmost r parity bits

In light of this choice, it is useful to further subdivide each row of blocks $B_{i,j=0..T-1}$ into its $n - r$ leftmost columns and its r rightmost columns, as illustrated in Figure V.3. The code rate of the code is $R = (n - r)/(n)$, while the overhead is $r/(n - r) \times 100\%$.

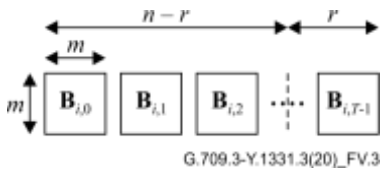


Figure V.3 – Subdivision of row of blocks $B_{i,j=0..T-1}$ into its $n - r$ leftmost columns and its r rightmost columns

The leftmost $2n - r$ bits in W are divided into two sub-blocks of which the leftmost contains n bits and the rightmost contains $n - r$ bits, as illustrated in Figure V.4.

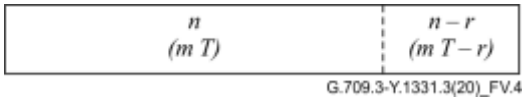


Figure V.4 – Subdivision of the leftmost 2*n*-*r* bits into its leftmost *n* information bits from previous columns and *n* – *r* information bits from a current row

The leftmost n (= m T) bits of W, referred to as the "front bits", are assembled from T different columns in T (out of $2T$) different even or odd previous rows of $m \times m$ matrices, as illustrated in Figure V.5.

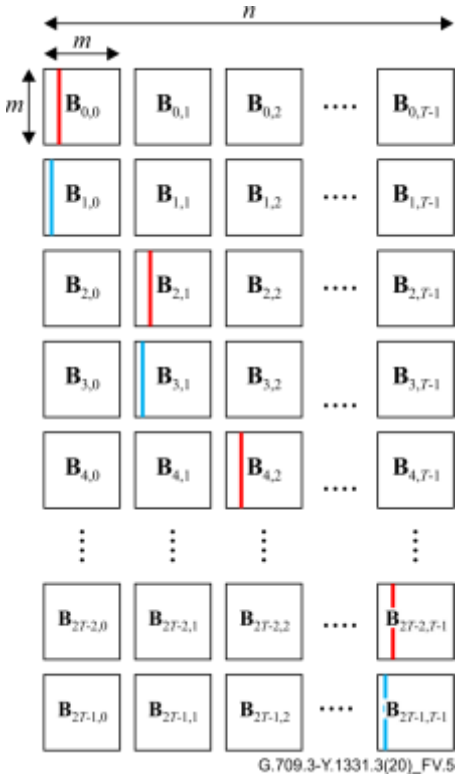


Figure V.5 – Origin of leftmost *n* symbols of a code word W

The rightmost $n - r$ *(= m T – r)* bits of the information bits in W, referred to as the "back bits", are assembled from the first $n - r$ *(= m T – r)* bits of one row $2T + 2G$ or $2T + 2G + 1$ of the current matrices, as illustrated in Figure V.6. Here, the $2G$ rows (i.e., $2T$, $2T$+1, .., $2T$+$2G$-1) of blocks located below the "front bits" and above the "back bits" of a given constituent code word are so called "guard blocks". Their purpose is to allow pipelined implementations.

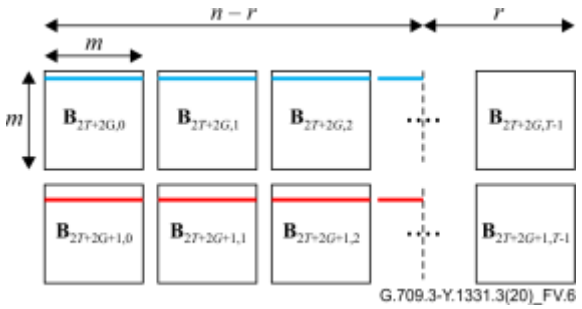


Figure V.6 – Origin of rightmost *n* – *r* symbols of the information positions in a code word W

For simplicity, we define the $m \times n$ matrix $M_i \triangleq [B_{i,0}\, B_{i,1} \ldots B_{i,T-1}]$. It is useful to further define its $n - r$ leftmost columns as $M_{(i,L)}$ and its r rightmost columns as $M_{(i,R)}$. Prior to encoding, matrices M_0 to $M_{2T+2G-1}$ (i.e., blocks $\{ B_{i,j} \mid i=0,1,\ldots,2T+2G-1,\ j=0,1,\ldots,T-1 \}$ are initialized to a reference state (e.g., matrices M_0 to $M_{2T+2G-1}$ could be initialized to the all-zeros state; note that the specific choice of initialization is unimportant, since the decoder is required to "bootstrap" itself from an unknown starting state, i.e., the decoder cannot exploit any knowledge of the reference state). Next, $m(n-r)$ information bits are stored in the $m \times (n-r)$ matrices $M_{(2T+2G,L)}$ and $M_{(2T+2G+1,L)}$, then the values of the $m \times r$ matrices $M_{(2T+2G,R)}$ and $M_{(2T+2G+1,R)}$, are calculated as follows:

1) Form the $m \times (2n-r)$ matrix $\Lambda_L = [B_{1,0}{}^T \; B_{3,1}{}^T \ldots B_{2T-1,T-1}{}^T \; M_{(2T+2G,L)}]$. Form the $m \times (2n-r)$ matrix $\Lambda_L' = [B_{0,0}{}^T \; B_{2,1}{}^T \ldots B_{2T-2,T-1}{}^T \; M_{(2T+2G+1,L)}]$.

   (NOTE – $B_{i,j}{}^T$ is the transposed matrix of $B_{i,j}$.)

2) The bits of $M_{(2T+2G,R)}$ are then computed such that each of the rows of the matrix $\Lambda = [B_{1,0}{}^T \; B_{3,1}{}^T \ldots B_{2T-1,T-1}{}^T \; M_{(2T+2G,L)} \; M_{(2T+2G,R)}]$ is a valid code word of W. That is, the bits in the i-th row of $M_{(2T+2G,R)}$ are exactly the r parity bits that result from encoding the $2n-r$ "information" bits in the i-th row of Λ_L.

   The bits of $M_{(2T+2G+1,R)}$ are then computed such that each of the rows of the matrix $[B_{0,0}{}^T \; B_{2,1}{}^T \ldots B_{2T-2,T-1}{}^T \; M_{(2T+2G+1,L)} \; M_{(2T+2G+1,R)}]$ is a valid code word of W. That is, the bits in the i-th row of $M_{(2T+2G+1,R)}$ are exactly the r parity bits that result from encoding the $2n-r$ "information" bits in the i-th row of Λ_L'.

Generally, the relationship between successive blocks in an Open FEC code satisfies the constraints imposed by the following relation:

For any $t \geq 2T+2G$, each of the rows of the $m \times 2n$ matrices

$[B_{t+1-2T-2G,0}{}^T \; B_{t+3-2T)-2G,1}{}^T \; B_{t+5-2T)-2G,2}{}^T \ldots B_{t-3-2G,T-2}{}^T \; B_{t-1-2G,T-1}{}^T \; M_t]$ for t even

$[B_{t-1-2T)-2G,0}{}^T \; B_{t+1-2T)-2G,1}{}^T \; B_{t+3-2T)-2G,2}{}^T \ldots B_{t-5-2G,T-2}{}^T \; B_{t-3-2G,T-1}{}^T \; M_t]$ for t odd

is a valid code word of W.

The rate of an Open FEC code is:

$$R = 1 - \frac{r}{n}$$

since encoding produces r parity symbols for every set of $n-r$ "new" parity symbols. Note that the related product code has rate:

$$\left(\frac{2n-r}{2n}\right)^2 = 1 - \frac{r}{n} + \frac{r^2}{4n^2},$$

which is greater than the rate of the Open FEC code. However, for sufficiently high rates, their rate difference is small, and furthermore, the Open FEC code outperforms a product code of the same rate.

Similarly, while the block length of the related product code is $4n^2$, the Open FEC codes are naturally unterminated (i.e., their block length is indeterminate), and thus admit a range of decoding strategies with varying latencies.

Finally, using arguments analogous to those used for product codes, a component code W with minimum distance d results in an Open FEC code with minimum distance at least d^2.

V.2 Permutation function

For hardware-friendly design, it is usual to consider the side length m of the square blocks being a power of two. To improve the performance of the Open FEC code described in previous section, the permutation process is introduced in the code structure. Consider a permutation function f on the $m \times m$ matrix B, where the element in the i-th row and j-th column of the $m \times m$ matrix f(B) is the same as the entry in i-th row and $(j \wedge i)$-th column of B. Here $(j \wedge i)$ represents the number with a binary representation equal to the bit-wise "exclusive or" of the binary representations of the numbers j and i. The encoding process is modified as follows.

Without loss of generality, it is supposed that matrices $\mathrm{M}_{t-2T-2G}$ to M_{t-1-2G} are available prior to encoding for the t-th row of matrix M_t, for $t \geq 2T+2G$. Consider a $m \times n$ matrix $\widetilde{\mathrm{M}} = [\widetilde{\mathrm{B}}_0\, \widetilde{\mathrm{B}}_1 \ldots\, \widetilde{\mathrm{B}}_{T-1}]$, where $\widetilde{\mathrm{B}}_j$ is the j-th $m \times m$ block. It is useful to further define its $n - r$ leftmost columns as $\widetilde{\mathrm{M}}_{(\mathrm{L})}$ and r rightmost columns as $\widetilde{\mathrm{M}}_{(\mathrm{R})}$. The current $m(n - r)$ information bits are first stored in the $m \times (n - r)$ matrix $\widetilde{\mathrm{M}}_{(\mathrm{L})}$. Next, the values of the $m \times r$ matrix $\widetilde{\mathrm{M}}_{(\mathrm{R})}$ are calculated as follows:

1) Form the $m \times (2n - r)$ matrix $\Lambda = [f(\mathrm{B}_{t-2T-2G,0}{}^{\mathrm{T}})\ f(\mathrm{B}_{t-2T-2G+1,1}{}^{\mathrm{T}})\ f(\mathrm{B}_{t-2T-2G+2,2}{}^{\mathrm{T}}) \ldots f(\mathrm{B}_{t-2G-2,T-2}{}^{\mathrm{T}})\ f(\mathrm{B}_{t-2G-1,T-1}{}^{\mathrm{T}})\ \widetilde{\mathrm{M}}_{(\mathrm{L})}]$.

2) The bits of $\widetilde{\mathrm{M}}_{(\mathrm{R})}$ are computed such that each of the rows of the matrix $[\Lambda\ \widetilde{\mathrm{M}}_{(\mathrm{R})}]$ is a valid code word of W.

Then the bits of $m \times n$ matrix M_t are computed such that $\mathrm{B}_{t,j} = f(\widetilde{\mathrm{B}}_j)$ for j=0,1,…,T-1, where $\mathrm{B}_{t,j}$ and $\widetilde{\mathrm{B}}_j$ are the j-th $m \times m$ block of M_t and $\widetilde{\mathrm{M}}$, respectively.

V.3 Decoding an open forward error correction code

Any of the iterative algorithms designed for turbo decoding of Product Codes can easily be adapted to decode open FEC code words.

For use with iterative decoding, observe that the bits in a square block row will all have been decoded as front bits in later constituent code words after $2T+2G+1$ rows of blocks have been decoded. Specifically, bits in square block row M_t will all have been decoded as front bits by the time block row $\mathrm{M}_{t+2G+2T}$ has been decoded. It then makes sense to decode the constituent code words in block row M_t again.

Bibliography

[b-ITU-T G.959.1] Recommendation ITU-T G.959.1 (2018), *Optical transport network physical layer interfaces*.

[b-ITU-T G-Sup.58] ITU-T G-series Recommendations – Supplement 58 (2018), *Optical transport network module framer interfaces.*

[b-ITU-T G.7712] Recommendation ITU-T G.7712/Y.1703 (2010), *Architecture and specification of data communication network*.

[b-Feltström] Alberto Jiménez Feltström & al., Braided Block Codes, IEEE Tr. On Information Theory, June 2009.

[b-Tanner] Robert M. Tanner, *Error Correcting Coding systems*, Fig 24 and Col 31, US Patent 4,295,218, 1981.

[b-Zigangirov] Kamil Zigangirov & al., *Encoders and Decoders for Braided Block Codes*, ISIT 2006, Seattle, 2006.

ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS, NEXT-GENERATION NETWORKS, INTERNET OF THINGS AND SMART CITIES

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
General	Y.1000–Y.1099
Services and applications	Y.1100–Y.1199
Architecture, access, network capabilities and resource management	Y.1200–Y.1299
Transport	**Y.1300–Y.1399**
Interworking	Y.1400–Y.1499
Quality of service and network performance	Y.1500–Y.1599
Signalling	Y.1600–Y.1699
Operation, administration and maintenance	Y.1700–Y.1799
Charging	Y.1800–Y.1899
IPTV over NGN	Y.1900–Y.1999
NEXT GENERATION NETWORKS	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Enhancements to NGN	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Computing power networks	Y.2500–Y.2599
Packet-based Networks	Y.2600–Y.2699
Security	Y.2700–Y.2799
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999
FUTURE NETWORKS	Y.3000–Y.3499
CLOUD COMPUTING	Y.3500–Y.3599
BIG DATA	Y.3600–Y.3799
QUANTUM KEY DISTRIBUTION NETWORKS	Y.3800–Y.3999
INTERNET OF THINGS AND SMART CITIES AND COMMUNITIES	
General	Y.4000–Y.4049
Definitions and terminologies	Y.4050–Y.4099
Requirements and use cases	Y.4100–Y.4249
Infrastructure, connectivity and networks	Y.4250–Y.4399
Frameworks, architectures and protocols	Y.4400–Y.4549
Services, applications, computation and data processing	Y.4550–Y.4699
Management, control and performance	Y.4700–Y.4799
Identification and security	Y.4800–Y.4899
Evaluation and assessment	Y.4900–Y.4999

For further details, please refer to the list of ITU-T Recommendations.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	**Transmission systems and media, digital systems and networks**
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling, and associated measurements and tests
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	**Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities**
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2023